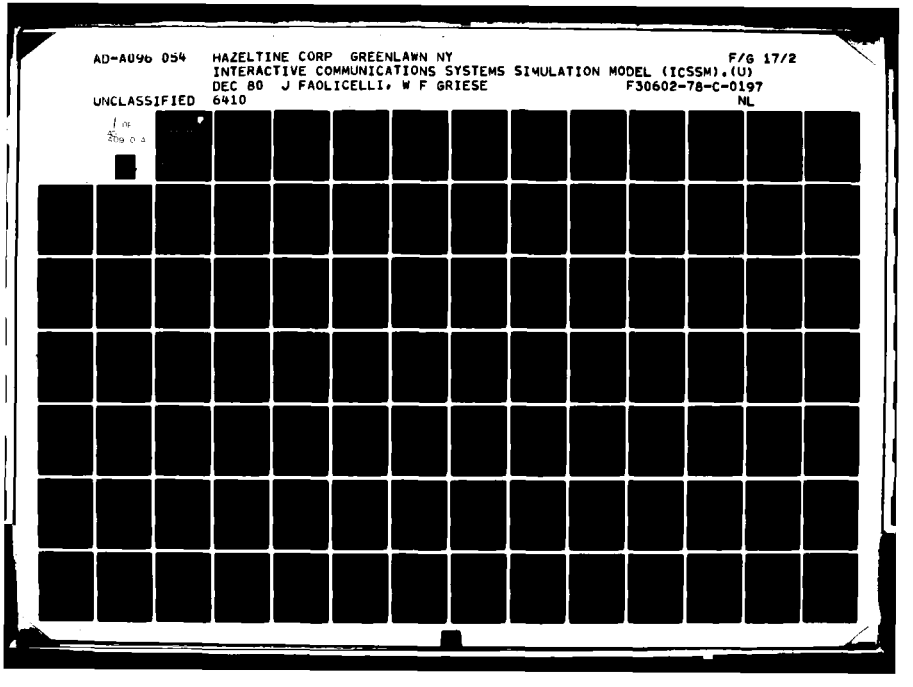

AD-A096 054
HAZELTINE CORP GREENLAWN NY
F/G 17/2
INTERACTIVE COMMUNICATIONS SYSTEMS SIMULATION MODEL (ICSSM).(U)
DEC 80 J FAOLICELLI, W F GRIESE
F30602-78-C-0197
UNCLASSIFIED
6410
NL

RADC-TR-80-373
Final Technical Report
December 1980

INTERACTIVE COMMUNICATIONS SYSTEMS SIMULATION MODEL (ICSSM)

Hazeltine Corporation

Joseph Paolicelli
William F. Griese

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441

This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-80-373 has been reviewed and is approved for publication.

APPROVED: Peter K. Leong

PETER K. LEONG
Project Engineer

APPROVED: Fred I. Diamond

FRED I. DIAMOND, Technical Director
Communications and Control Division

FOR THE COMMANDER: John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (DCLF) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return this copy. Retain or destroy.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19 REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: RADC-TR-80-373	2. GOVT ACCESSION NO.: AD-A096 034
3. RECIPIENT'S CATALOG NUMBER:	
4. TITLE (and Subtitle): INTERACTIVE COMMUNICATIONS SYSTEMS SIMULATION MODEL (ICSSM)	5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report. July 78 - Apr 80
6. PERFORMING ORG. REPORT NUMBER: 6410	
7. AUTHOR(s): Joseph Paolicelli, William F. Griese	8. CONTRACT OR GRANT NUMBER(s): F30602-78-C-0197
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Hazeltine Corporation, Greenlawn NY 11740	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 62702F, 45192011
11. CONTROLLING OFFICE NAME AND ADDRESS: Rome Air Development Center (DCLF), Griffiss AFB NY 13441	12. REPORT DATE: December 1980
	13. NUMBER OF PAGES: 153
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Same	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: Peter K. Leong (DCLF)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Point-to point Communications Design & Analysis Aids
Communication Systems Modeling
Multi-port Block Diagram Modeling
Interactive Computer Simulation
Time-step/Event-step Simulation
Monte Carlo Simulation
User-System Interface
Modular Software Structure
Re-entrant Modules

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The Interactive Communication Systems Simulation Model (ICSSM) developed for Rome Air Development Center is capable of simulating a point-to-point communication system including its functional elements, components, propagation effects, and transmission media. The ICSSM is a flexible, expandable, sophisticated and easy-to-use computerized means to develop or configure communication system specific simulation models; specify and validate system requirements; evaluate new techniques and assess (Cont'd)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED 406972

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Item 20 (Cont'd)

cont

the performance of existing and proposed conventional and ECCM communications systems and equipment.

The ICSSM's preconfigured programming structure frees the analyst from the burden of constructing a special simulation framework for each model effort, thus permitting him to concentrate on the model formulation itself. Further, the analyst may benefit from the legacy of previous modeling via the ICSSM library of communication model elements which are supported by computerized tutorials and guides.

The development of the initial ICSSM concentrated on efficient system structure, a generalized simulation capability and on making the system easy to use. The ICSSM increases in utility with continued use as additional modeling elements are incorporated in the expandable library.

CONTENTS

CONTENTS (Cont)

CONTENTS (Cont)

ILLUSTRATIONS

ILLUSTRATIONS (Cont)

TABLES

ACKNOWLEDGEMENTS

Many have contributed with persistence, skill and imagination to the success of the ICSSM System project. In particular, Hazeltine Corporation was pleased to have, throughout this project, the constant support, able guidance, and technical assistance of P. Leong and D. McAuliffe of the Rome Air Development Center of USAF. Credit is due J. Angell, I. Gerry, K. Mester, J. Paolicelli, F. Torre, W. Fifer and J. Sutton of Hazeltine Corporation and S. Ginsberg and A. Novick of Mission Sciences Corporation, who, with the encouragement of W. Griese, Head of Hazeltine's Systems Analysis and Evaluation Laboratory, and R.H. Cope, Vice President for Research, Hazeltine Corporation, successfully completed the developments reported on here.

EVALUATION

The final report on this effort provides descriptions of the functions, system structure, program construct, and utilities of the ICSSM, as it was conceived, designed and implemented. Detailed data on the ICSSM are contained in documents such as Functional Description, System/Subsystem Specification, Program Specification, User/Analyst Manual, and Program Maintenance Manual which have been prepared IAW DoD ADS Documentation Standards 4120.17-M. This effort is the initial implementation portion of the ICSSM development project, which in turn is an on-going project, part of RADC Technical Planning Objective (TPO) 4B3, "Communications/C^3 System Design and Analysis."

The value of this effort is considered significant when viewed against the ultimate goal of this program. The ultimate goal is to provide Air Force scientists and engineers with a fully supported, flexible, expandable, sophisticated, and easy-to-use computerized tool to 1) design, specify and validate requirements, 2) evaluate technique effectiveness, and 3) assess system/link performance, of existing and proposed conventional and ECCM communications equipment/capabilities. The reported work resulted in a basic operational computerized software system called ICSSM, which is verified, well documented and consists of all the desired functional characteristics of the envisioned ICSSM to a very large extent. So much so that it can easily accommodate any conceivable future growths (both system- and application-related enhancements), and be responsive to the modeling, simulation and analysis requirements generated by new communications system development needs. This is mainly attributed to the fact that ICSSM is both machine- and application-independent as far as software code conversions and system modeling/analysis, respectively, are concerned. Its over 8000 lines of FORTRAN IV source codes have

been written IAW ANSI Standards FORTRAN for conversion ease. Its multi-input, multi-output block diagram structure with flexible interconnections allows modeling of practically any system (even though intended for communications systems/links) that are decomposable to interconnected functional blocks. Furthermore, there is, as an integral part of ICSSM, an application library to facilitate storage, and access of all application software, be it simulation/modeling modules, analysis subroutines, or computer graphics subroutines. Development of any new application software needs only to comply with the user's modeling/simulation requirements, the input/output requirements set for any ICSSM Library module, and to ensure that the modules be re-entrant. Thus the ICSSM as stands now is a baseline modeling and simulation tool that can "grow" with the communication systems design and analysis requirement put upon it, and will thence become more and more responsive to follow-on requirements.

Two practical problems have also been modeled and simulated by using ICSSM, primarily for validating the capability and utility of ICSSM in solving such problems with the required accuracy levels in reasonable computation times. The results are documented in Section 4 of this report. Basically, the overhead processings required to handle and control the models are but small portions (about 10%) of the simulation processes. That is considered very efficient especially in view of the high degree of flexibility built into the ICSSM software structure. The accuracy of results and speed of simulation, because of their dependency on specific computer host hardware and software (in this case the Honeywell H-6180 MULTICS system), and the utility and application programs available or provided, are considered reasonable. Many possibilities for improvement have been considered and some will be further developed to enhance accuracy and simulation speed (see Section 5).

Based on the results of this effort, a follow-on effort is being conducted 1) to augment the ICSSM with user-oriented functions and system support utility functions, 2) to provide documentation suitable for competitive and compatible development of future system improvement and application software, 3) to expand the existing application software library and 4) to model, simulate, and analyze the synchronization acquisition mode of a candidate Air Force tactical spread spectrum modem.

Peter Leong

PETER K. LEONG
Project Engineer

SECTION 1

IMPETUS AND BACKGROUND FOR ICSSM DEVELOPMENT

This Section provides a sketch of the background and history of the ICSSM development project and outlines the functional requirements and design objectives of the ICSSM System.

1.1 PURPOSE

This document constitutes the Final Technical Report (FTR) on the Interactive Communication System Simulation Model (ICSSM) developed by Hazeltine Corporation for the Rome Air Development Center under Contract No. F30602-78-C-0197.

The purpose of this FTR is to summarize the results achieved in performing the work of ICSSM development, and to make recommendations for future enhancements to the ICSSM system.

This FTR is organized into 5 Sections which describe the motivations, rationale, implementation, results, and recommendations for ICSSM.

1.2 SPON$ORSHIP AND PROJECT GUIDANCE

ICSSM System development was undertaken in response to Requirements described in Request for Proposal F30602-77-R-0185 (PR#C-7-2027), responded to by Hazeltine Corporation via "Technical Proposal for Interactive Communications System Simulation Model" Hazeltine Report No. 6326, dated December 14, 1977.

ICSSM was designed and developed under the guidance of RADC/DCLF personnel, in accordance with the items of description in the Statement of Work for PR#C-7-2027. The development, test, and documentation of ICSSM are in accord with extant, applicable Air Force and DoD guidelines, standards, and procedures, including:

- RADC Computer Software Specification CP07877796100B
- DoD Manual 4120.17.M for Automated Data Systems Documentation Standards

1.3 REFERENCES

Immediately relevant technical references which define, express, set the tone for some aspects of ICSSM design, or provide needed technical details of specific applications models selected to demonstrate ICSSM operation, are included in Appendix A. Hazeltine documents previously published in connection with the ICSSM project are also listed in Appendix A.

1.4 ICSSM SYSTEM DESCRIPTION AND CAPABILITIES

The ICSSM system consists of a software executive or control program, model specification and data reduction programs, plus an Applications Library of computerized modeling elements, for non-real-time computer simulation of point-to-point digital communication systems. ICSSM provides U.S. Air Force engineers and scientists with a powerful, simple-to-use, interactive simulation capability operating on a computer system at RADC.

1.5 BACKGROUND AND MOTIVATION FOR ICSSM DEVELOPMENT

The application of digital computer simulation to communication systems analysis and design has a long history. The efficacy of these efforts depends upon the software and hardware available at the time of their conception and execution, upon the state of the software engineering art, and upon the sophistication of the communication analyst's science and art at the time. Many of those simulation efforts could be said to have been successful, particularly when the then current states of computer science and communication science are considered. However, the recent literature and experience strongly suggest the overall characteristics of an improved communications system simulator:

- Sophisticated wideband and computation-intensive communication system designs are of current interest and will become more so. Thus, a simulator should operate at high speed to deal with the very large number of data samples that may be needed to characterize the operation of such communications systems.
- Anticipated advances in the communication systems art require that a simulator be flexible and adaptable if it is to service the analyst's needs in the future.
- The sophistication of modern communication systems, and the emphasis now placed on performance prediction in the design phase of these systems, requires that a simulator be accurate.
- The rapid growth of communication science, and the simultaneous needs to express new analytic and design ideas, and to explore the performance limits of communication system configurations operating under very adverse conditions require that a simulator be expressly general in its application potential, and to possess an extraordinary degree of faithfulness and validity in its results.
- The increasing specialization apparent in all fields of technical endeavor, including the computer and communication sciences, requires that a simulator be unobtrusive and easy to use, with apparent relevance to the communications analyst's view of the problem.

- The simulator should take advantage of, and anticipate, new capabilities in available computer equipment and operating software if it is to remain viable. Therefore the simulator must be transportable, employing generalized computational technique.

- The number of functioning communication/processing elements of the communications systems to be simulated, and the level of detail needed to explore the simulated system's performance are very variable, and can result in very large configurations or collections of computational or functional elements within a single model. The effective simulator must be elastic or expandable in configuration and simulation capability.

Thus, the requirements for a state-of-the-art communication system simulator are defined by the properties of:

- speed
- flexibility/expandability/adaptability
- generality
- accuracy/validity
- ease of use/relevance
- transportability

1.6 DESIGN OBJECTIVES FOR ICSSM

The design objectives for the entry-level ICSSM configuration are related directly to the requirements/properties disclosed in paragraph 1.6.

1.6.1 Speed as an Objective

Software overhead attributable to ICSSM control program operation should be less than 10% of the computational load. Simulation-model-specific process requirements should account for at least 90% of processing time for simulations. When host hardware is augmented by an array processor, ICSSM should provide a processing ratio of 1 sec of real time to 1 hour machine time for a selected system model of current interest, chosen for the detail of its representation and for its concomitant heavy computational burden.

1.6.2 Accuracy and Validity as Objective

ICSSM should be so designed that the validity and accuracy of the results of a modeling exercise is dependent solely on the faithfulness with which the User/analyst designs and implements the functional elements of the model, and not on the internal structure of ICSSM control/executive software.

1.6.3 Ease of Use and Relevance as Objectives

The ICSSM design should emphasize the natural analytic and mathematical constructs of the communications analyst, and should provide (interactive) input facilities couched in language and familiar-in-form or relatable to the conceptual needs of the analyst or communications engineer.

ICSSM should further provide a simple regiment for describing (and, where possible or appropriate, altering previously described) communications models, in computer-efficient form, with no unnecessary artifices which impede the analyst's intuitive "feel" for the model.

1.6.4 Transportability as an Objective.

ICSSM should be designed in the standard (ANSI) FORTRAN language and should avoid reliance on computer operating system capabilities or resources for its internal operation,

or on peculiarities or special features of particular computers to meet its objectives, unless these special features can normally be expected to exist on the computer systems likely or intended to provide host for ICSSM.

1.7 IMPROVEMENTS AFFORDED BY ICSSM.

The ICSSM design capabilities are directly relatable to extant User/analyst (U/A) needs. The requirements envelope of paragraph 1.7 et seq defines a simulation capability that improves significantly over some existing simulation facilities, constituting a unique and novel confluence of desirable features. In brief, the improvements afforded by the ICSSM are:

a. Accuracy and applicability limited only by the ingenuity and modeling requirements of the User/analyst and by the capacities of the host computer.

b. Flexibility and adaptability which accommodates common design/analytic regimens in communications system engineering and naturally suggest or accommodate other regimens, some heretofore too impractical for common implementation.

c. Ability to model and examine portions of communication system designs, in greater detail, the remainder being modeled in lesser detail, completely in consonance with the User/analyst's actual interest and needs.

d. Ability to decompose a communication system model into convenient "pieces," which can be exercised sequentially and repeatedly, each time using the entire host computer resource.

e. Natural accretion and reuse of functional electronic modeling elements (via registration in the permanent ICSSM Library) so that the legacy of previous modeling efforts need not be lost.

f. Facilities for selecting and controlling the types and volume of output data generated and collected in accord only with the User/analyst's particular interests and needs.

g. Interactive facilities which preserve the User/analyst's "feel" for his model, and which render the presence of the simulation software unobtrusive, except where discipline is imposed to insure model correctness and insure efficient use of host computer resources.

h. Simulation time (end-to-end computer processing time) determined almost exclusively by the physical nature of the communication system being modeled, the detail with which the user/analyst (U/A) wishes to express the model and examine the model's performance, and the throughput capabilities of the host computer, and almost independently of the internal structure and requirements of the ICSSM control/executive software.

1.8 LIMITATIONS PRESENT IN ICSSM DESIGN

The generality and flexibility of the ICSSM design, and its emphasis on transparency to the User/analyst certainly relieve many limitations now presented by other simulation methods. Specific limitations arise or are imposed by host computer resources and by the needed structural disciplines for Library element design. ICSSM is designed for point-to-point digital communication system simulation. However, continuous waveform system simulation (eg, an analog FM system) can be handled but only in sampled-data form.

It is possible to exceed current ICSSM capabilities if simulations are attempted that effectively require more samples (computer data elements) within an ICSSM internally-defined sample/event packet than ICSSM can represent and still remain true to physical law (eg, it is possible to violate the sampling theorem).

When simulation study of the extremely rare event is required (eg, extremely small probability-of-error operating points), the basic random process for simulating, for example, noise corruption, may never be exactly suitable in that it will never be exactly random, no matter what statistic is of interest. Thus, it is possible to define a simulation which will never produce enough valid samples before termination of the simulation.

ICSSM has been designed with specific intention to use an Array Processor of the capabilities of the Floating Point Systems, Inc., Model AP120B. The Array Processor is required to meet the speed objectives of ICSSM. While ICSSM possesses software which emulates Array Processor operations (these are available in the Library), using these emulation facilities will adversely affect the speed of the simulation exercises themselves.

SECTION 2

ICSSM DESIGN RATIONALE

2.1 INTRODUCTION

This Section describes the rationale for ICSSM design. It addresses the modeling philosophy, the communications system representation problem, the principles of validation, and the key design features which help determine the ICSSM system configuration.

2.2 THE BLOCK DIAGRAM APPROACH TO COMMUNICATIONS SYSTEM MODELING

ICSSM design is based upon a "block diagram" approach to communications systems modeling. This approach is epitomized in figure 2-1. There, elementary functional processes employed in a communications system are represented by blocks. Each block represents a separate (mathematical) function which maps elements of a domain space (input port) into elements of a range space (output port). The sequence with which the functions are applied is indicated by connecting lines drawn from an output port of a function block to an input port of some (usually) other function block. The connections themselves represent mathematical identity operators.

The function blocks represent unilateral operations; while they may represent mathematically invertable operators, they are never considered in this light. Thus, in this approach, the inverse of a given function, if required, is always represented by a block (or blocks) representing the inverse per se.

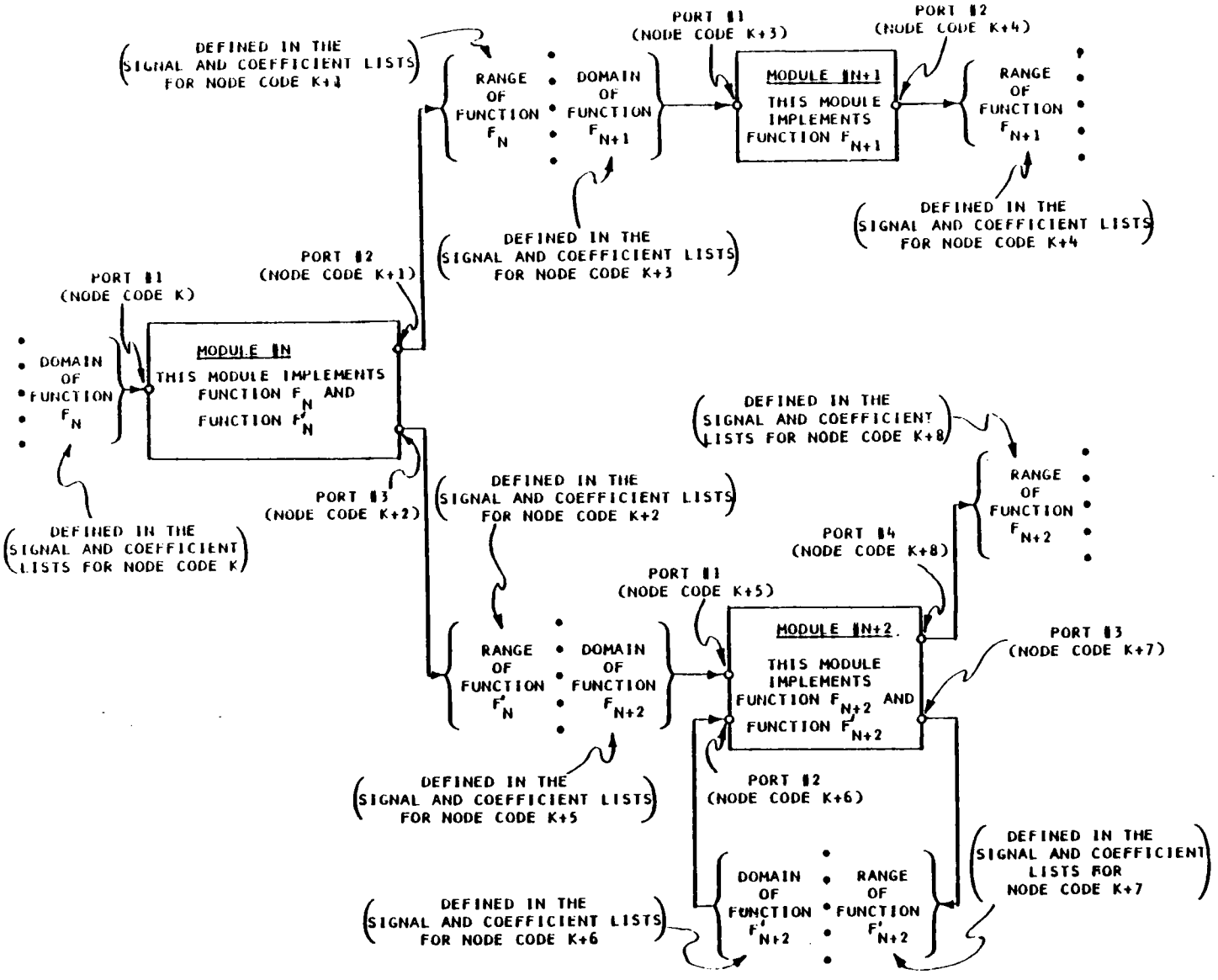


Figure 2-1. Block Diagram Approach to Communications System Modeling

2.2.1 Domains and Ranges in ICSSM Models

The elementary functional processes represented on figure 2-1 map particular domains into particular ranges. For example, a function may map a set of time-dependent voltage values (domain) into another set of time-dependent voltage values (range). Within the ICSSM system, the U/A has complete freedom to define the "meaning" of any domain or range space consistent with his modeling requirements. Particular elements (which could, in themselves be functions) from the desired domain or range sets are represented within ICSSM as (ordered) sequences of values of the dependent variable required, the nature of the corresponding (ordered) sequence of values of the independent variable being implied or defined by the nature of the associated elementary functional process.

For example, suppose a domain were to consist of (time-ordered, regularly-spaced) amplitude samples of a function representing a signal voltage. The domain would then be represented in ICSSM as a set of values (called Coefficients) arranged in time-ordered sequence and made available as input to the relevant elementary functional process manifested in a related ICSSM Applications Library module. The results of the functional processing would be generated (by the actions of the Application Library module/algorithm) as an ordered sequence of range values made available within ICSSM for subsequent processing by other modules.

2.2.2 Coefficient and Signal List Records in ICSSM

Within ICSSM, ordered sets of domain and range values are termed Coefficient Records and are managed, within ICSSM as Coefficient Lists. The Coefficient Records associated with a particular port of a particular module of the block diagram are identified within the ICSSM system by a combination of module number and port number from which a unique Node Code number is derived. The definition or

meaning ascribed to the elements of particular Coefficient Records depends on the associated Application Library module selected or specified by the U/A. The generation of Coefficient Records, and Node Codes, and the arrangement and control of the Coefficient Lists is transparent to the U/A.

Associated with a given domain or range set (ie, Coefficient Records) ICSSM employs Signal List Records containing values of descriptors, attributes, derived quantities, or describing/delineating quantities related to the associated functional module or Coefficient Record.

For example, if a Coefficient Record contained values of power associated with particular frequencies in the spectrum existing at a particular Node, the Signal List Record would perhaps contain values for the frequency spacing of the spectral values, and perhaps a measure of the total extent in frequency (ie, the bandwidth) represented by the Coefficient Record. Generally, associated with a Coefficient Record, one may need or find additional Signal List Record quantities. Some of the Signal List elements are "derived from" the Coefficient Record elements by integral operations (eg, a Signal List value of average power found associated with Coefficient Record entries representing voltage as a function of time). Sets of values constituting a Signal List Record are also identified by a Node Code number as with Coefficient Records. Management and manipulation of Signal List Records are transparent to the U/A.

2.2.3 Generality and Flexibility in ICSSM

ICSSM achieves generality and flexibility in its models in part through the use of the aforementioned Coefficient Record and Signal List Record concepts. Concomitantly some caution is called for in algorithm design for ICSSM Applications Library modules and in actual model formulation using ICSSM. Care must be exercised to confirm that the measurement units of the output (range) Coefficient

Record and Signal List Record elements for a given module are compatible with the units required by the input port (domain) units of the module to which it may be connected. If, for example, a given module produces a spectrum in its output Coefficient Record, and it is connected to another module which uses amplitude values of a bit stream in its input Coefficient Record, these modules would be incompatible. However, ICSSM would not warn the U/A of such an impending incompatibility during the ICSSM-aided model configuration steps. Such caution must be exercised by the U/A so that he knows the nature of the modules he seeks to interconnect in his model.

For some models, simulation may be performed without requiring Coefficient Records and Lists at all. In such cases, the algorithmic forms for the Applications Library modules constituting the model address more general properties or attributes of simulated communications signals, and not the signals themselves. By this means, models may be constructed which operate at levels of generality or abstraction other than that represented by bit-by-bit or signal-sample-by-signal-sample processing.

2.3 REPRESENTATIONAL AND ANALYTIC PROBLEMS IN COMMUNICATIONS SYSTEM MODELING

The ICSSM design is strongly influenced by the current and conceivable future states of the communication analyst's art. A brief discussion of the relevant issues will clarify the role of simulation in the practice of this art, and will isolate those themes which most strongly impact ICSSM design, particularly with regard to the design of functional elements (ie, Applications Library modules) employed in ICSSM.

The communication modeler's implicit assumption is that the mathematical framework of the communication art is somehow bounded. This assumption may be justified on several bases. For example, we may consider that all functions (ie, signal representations) employed in the art are square-integrable (Lebesgue sense). Thus, all operators on such functions must map the space L_2 into L_2. This already restricts the communication theorist's consideration substantially. Further, all functions suitable for a simulation representation must be computable (Davis), thus further restricting the class of functions to be considered. These and similar mathematical assertions may be too general for practical use here, but they do provide general assurances. Yet, a large body of electronic practice already exists which, from another point of view, also allows the assertion that the communication design discipline is far more bounded than the literature might indicate. To justify this claim, it is useful to briefly examine one particular class of problems. This example class will also motivate some general observations about the nature of the overall modeling and simulation problems for the communication analyst.

2.3.1 A Problem Example

The selected examples are characterized by the presence of additive interference constituting a sum of sinusoids - the multiple sinusoid interference (MSI) case. The central limit theorem does not apply here so that a gaussian assumption is invalid. Instances of this class include: co-channel interference dominated by a few (non-spread-spectrum) signals; intersymbol interference; self-interference in a frequency-time hopped (FTH) CDMA regime; and "multiple CW" jamming.

For the most part the literature (eg, [9] - [21]) has dealt with this interference type in the context of receiver designs optimum only for additive white gaussian noise (AWGN). The various analyses can be divided into two approaches: (1) numerical methods ([9] - [15]) and bounding approaches ([16] - [21]).

The numerical methods include series methods, the gaussian quadrature method, and the direct averaging method. When the vector of random variables representing the MSI has certain properties (which usually occur in practice) all these methods converge. However, the pertinent expressions have not been evaluated exactly. Truncations seem unavoidable, but for each method, the consequent errors are bounded and vanish in the limit. Yet, simple and accurate expressions seem practically impossible to obtain. No ranking or general critical comparison is available for these methods. Unfortunately, the current numerical methods do not consider many practical effects (eg, the influence of the MSI on carrier and timing recovery or the presence of system non-linearities).

The bounding approaches trade accuracy for a reduction in analysis complexity. These approaches fall into two classes: those using the Chernoff bound and those using the Maximizing Distribution bound. Results based on the Chernoff bound grow more accurate [16] - [18] as the signal-to-interference ratio increases. Perhaps the greatest virtue of the Chernoff bound is the insight it furnishes as to when it is reasonable to treat MSI as equivalent to AWGN of the same average power.

The Maximizing Distribution bound has been a most successful MSI analysis approach. The results obtained, [19] - [21], permit families of curves to be developed and stored as a table. The bound is tight enough for most cases of interest. However, the same practical effects are ignored as in the numerical methods. Thus, for more detailed works a regime using a combination of analysis and simulation computation is required.

Very little [22], [23] has been accomplished in identifying the form of the optimum receiver when both MSI and AWGN, are present. The meager results ignore multiplicative interference, and other practical effects.

2.3.2 The Status of Communications Analytic Theory

The above example illustrates characteristics of the communications discipline in general:

(1) the literature consists primarily of alternative analyses for a limited set of basic problem.

(2) typically, one of the alternatives provides the best approach; other alternatives merely augment the insights derived from the numerically best approach.

(3) invariably, the analyses fail to treat many of the factors relevant to the real problem from which the modeled problem is abstracted.

(4) the literature rarely addresses the derivation of structures optimum for real problems. With few exceptions, those constructs employed are optimum for gaussian statistics.

(5) at present, communication system design is based on gaussian-optimum constructs. The non-gaussian constituents are then accommodated by a combination of adaptive processing and ad hoc design.

Point (1), above, implies that the discipline of communication systems is relatively well-bounded, an attribute of fundamental importance to the viability of the ICSSM Applications Library concept.

Point (2) reinforces point (1), and emphasizes a major merit of the ICSSM: the common framework permits alternative design approaches to be simulated and compared so that inferior designs can be discarded.

Point (3) reflects the remarkably limited power of "applied mathematics" in dealing with the complex problems that arise in modern communication systems. Communication system design will ignore in most cases, the goal of completely analytical solutions, in favor of simulation as a major analytic/computation tool.

2.3.3 Merits and Potentials of Simulation

A fundamental merit of ICSSM is that, by assuming much of the computational burden, it permits the communication theorist/analyst to devote more time and ingenuity to correcting the pervasiveness of the gaussian ass-mption, as cited in point (4).

Consider, for example the block diagram of figure 2-2 which describes a broad class of communications link designs. If the application is digital communications (as in DCS-LOS) the modulation is most likely based on a signal-space geometry which minimizes (consistent with channel bandwidth limitations) the cross-correlation among an M'ary signal set. This approach is optimum for the AWGN channel model, but is generally sub-optimum for colored noise. For non-gaussian statistics, it is patently sub-optimum.

Consistent with the gaussian noise premise, the demodulation probably embodies matched filtering or correlation, to maximize the signal-to-noise ratio of the decision statistic.

Similarly, for analog parameter communication (eg, Pseudo-Noise Conferencing Modem) the demodulation invariably is based on a gaussian assumption for the interference random processes. The applicable science in this case is estimation theory wherein the statistics for both the information process and interference are relevant.

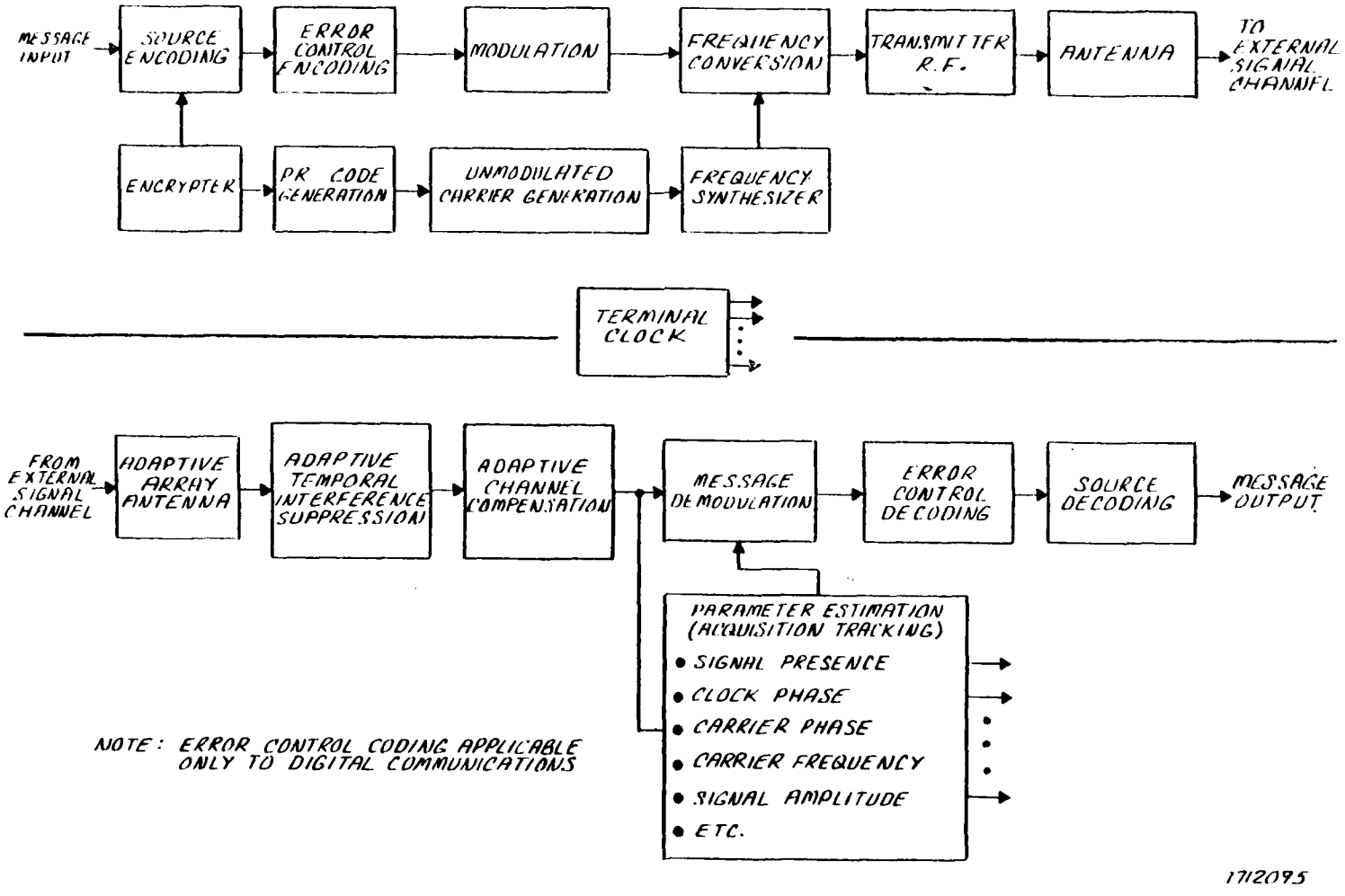


Figure 2-2. Block Diagram for Illustrative Communication System

The fidelity criterion selected (primarily because of tractability) is either the minimization of the mean-square estimation error (MME) or the minimization of the error variance for an unbiased estimate.

A state variable formulation of the problem permits the introduction of continuous Markov processes, but tractability invariably requires the assumption of a Gaussian-Markov process and usually dictates a linearization of the problem. This leads to the employment of phase and frequency locked loops, which are (essentially) a linearization of the Kalman filter concept, itself optimum for linear estimation of a gaussian process in gaussian noise interference. These considerations carry over intact to the estimations of phase, frequency, time, in digital demodulation, as in figure 2-2.

The limited number of constructs available to the communication link designer promotes the manageability of the ICSSM Applications Library concept. By a thorough modeling of relatively few generic constructs, a very large portion of the modeling required in communication system design in the foreseeable future can be addressed.

2.3.4 Aspects of Communication Model Evaluation

The relative paucity of design constructs provided by the theory for communication link design derives from the intractability of expressions for optimum designs.

Analytical tractability also limits the complexity of performance criteria. To a great extent this derives from inherent limitations of the underlying decision theory. In practice, decision criteria reduce to a few basic theories (eg, "minimization of symbol error probability" in digital communications, and "unbiased minimum error variance" for analog communications).

The designer often force-fits the aforementioned simple criteria to the actual problem. For a digital signaling design (as in DCS-LOS) this "force-fitting" manifests itself in the artifice of source encoding voice waveform into a sequence of "information bits." As a consequence, the quality of input to the source decoder is judged in terms of bit-error statistics.

Typically, digital communication design is based on the theoretical criterion of minimizing the probability of bit error, which is merely one of an infinite number of statistics characterizing bit-error patterns.

Simulation of the system may permit the study of bit-error statistics other than bit-error probability and may reveal the existence of channel memory. If this memory effect degrades actual performance, subsequent design iterations may include elements to counteract it. Simulation may disclose that the actual clustering of bit errors permits a relaxation on the required bit-error probability (for instance, speech intelligibility is known ([24], [25]) to be highly resistant to bit-error bursts of certain durations).

In both digital and analog communication system design, unexpected or "anamolous" performance degradation is often traceable to higher-order error statistics ie, other than the average bit-error rate or the average power of the voice-waveform estimation error. Significant degradation (if not outright failure) results from design concepts that apply simple performance criteria and idealized channel models to complex source/sink requirements and real channel characteristics. The anamolous behavior invariably involves statistics other than that chosen for optimization in the theoretical design.

Analyses which consider merely the theoretically optimum criteria often spawn designs which succeed on paper and fail in operation. Simulations can uncover the unexpected sources of degradation so that they can be properly mitigated in follow-up design iterations. Accordingly the ICSSM system is structured to permit access to any information node in the link model formulation, and not merely to preselected performance measures.

2.4 ACCURACY AND VALIDITY IN SIMULATION

Ideally, the accuracy of mathematical calculation in a simulation should be limited only by the word size, internal coding, and arithmetic capability of the host computer.

The validity of simulation results should depend only upon the validity of the functional elements employed within the formulated model and this, in turn, should depend upon: (a) the availability of algorithmic representations of the modeled processes; (b) the ingenuity and skill of the User/analyst in defining the performance limits and computational analogs manifested in simulation elements; and (c) the nature of the validation principles chosen.

Validation principles can be chosen from among several possibilities. Figure 2-3 shows an abstract representation of the conceivable modeling processes which need be considered. There the relationships among the "versions of reality" which exist in the communications system/simulation system setting are depicted. Validation consists of determining the fidelity of any of the transformation systems T_{AI}, T_{AS}, and T_{IS}.

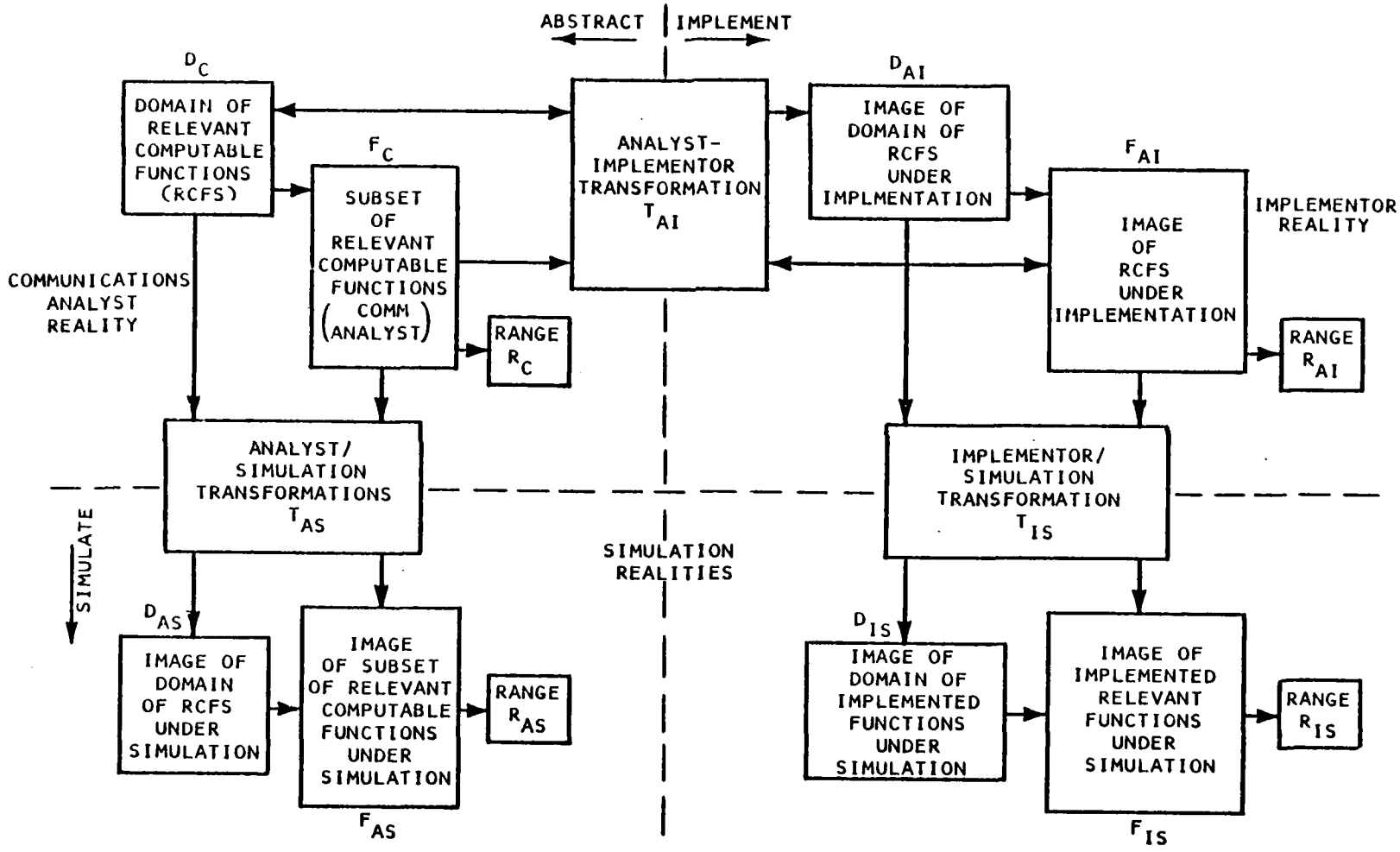


Figure 2-3. Versions of Reality

The "fidelity" can be measured by determining how "close" a certain result in one of the output range spaces lies with respect to its range pre-image under the given transformation, given that the image of a corresponding domain element in the output domain space lies specifically close to its domain pre-image. Figure 2-4 portrays the described simulation modes as they bear on the fidelity relations. There, validation is perceived and protrayed as "comparisons" between domain/range pairs in one reality and the corresponding pairs in another reality. Validity is also determined by the metrics or "measures of closeness" chosen.

A simulator should be designed to be independent of both the metrics employed and the simulation mode selected. Validity then must address the fidelity of modeling correspondence. The simulator's internal control/executive software should provide a facility to effect the mappings T_{AS} or T_{IS} but should be neutral with respect to their validity.

2.5 METHODS OF USING ICSSM FOR COMMUNICATIONS SYSTEM SIMULATION

The ICSSM system can simulate the performance of multiple-element communications systems. An ICSSM "target-simulation model" (TSM) can be fashioned to incorporate all the traditional elements of point-to-point links in one model: message source, coding and modulation steps; antenna, propagation and channel processes; receiver front end; demodulation and decoding processes; and message transduction. However, implementation limitations can restrict the number of different Applications Library modules employed in a TSM. The number of functional elements which can be employed, the number of interconnections permitted, the time available for exercising a TSM all have limits. However, ICSSM is capable of alternative styles of usage which can surmount most of the model-size limitations. Alternative methods of using ICSSM are discussed in the paragraphs below.

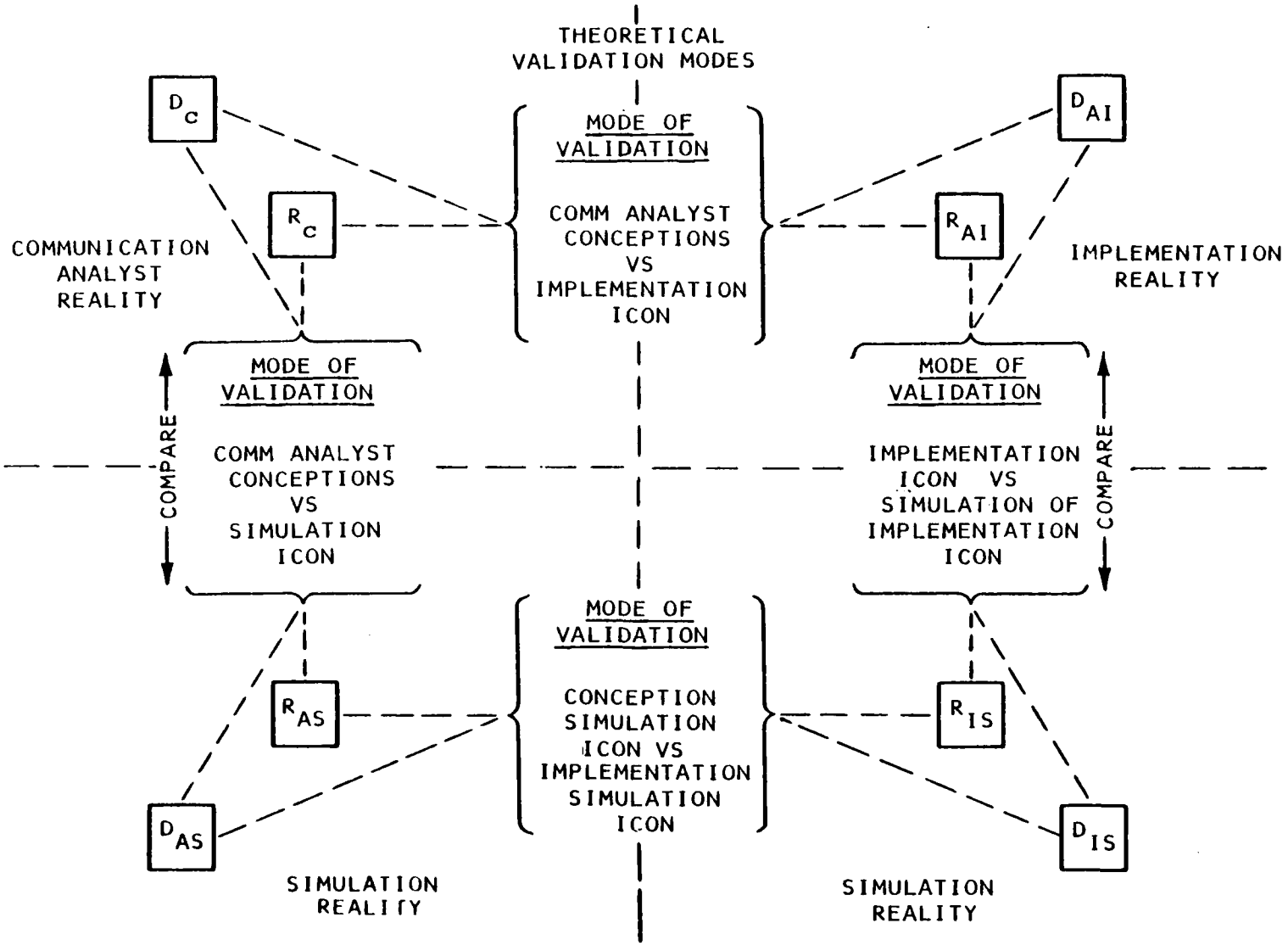


Figure 2-4. Theoretical Validation Modes

2.5.1 Basic Method of ICSSM Use

ICSSM consists of three main computer-based elements. These are: The pre-simulation "Configurator" (MC Subsystem); the simulation model Exercisor (ES Subsystem); and the post-simulation "Output Processor" (PP Subsystem). These elements are supported by the ICSSM Applications Library and by an auxiliary processor specifically designed for ICSSM Applications Library maintenance.

The ICSSM Applications Library contains many functional modules (sub-programs) which can be used in communication model construction. Access to this collection of modules is provided through a Library Directory and a Module Description File used mainly during Model Configuration. The selected functional modules (eg, envelope detector, signal generator, encoder, etc) are incorporated into the TSM during a pre-compile step. The ICSSM Applications Library also contains modules and program elements which simulate certain electronic test equipment and certain communication system testing/evaluation methods (eg, a module which computes bit-error-rate). TSM output data display/reduction processes are performed in the output data processing step either through the use of data reduction programs and subroutines which are available from the ICSSM Applications Library, or through the use of plotting, printing, display, and processing software from the host computer's standard support software.

The basic method of ICSSM usage involves specifying/configuring the entire communications system model, submitting the resultant TSM for execution, and examining the resultant output data.

2.5.2 Iterative or Closed Loop Use of ICSSM

The diagram of figure 2-5 suggests that the U/A employs ICSSM to complete a closed loop of methods, manipulations, and actions. This employment style can be extended to repeated or iterative use of the ICSSM facilities. This is consonant with the design or analysis of advanced communications systems since mathematical/analytic methods are often inadequate to characterize a system completely or precisely (refer to paragraph 2.3). ICSSM is designed to accommodate the iterative method: TSM configuration specifications created by the (interactive) cooperation among the U/A, the ICSSM configuration program, and the ICSSM Library Directory "help" files, are captured and formatted in an intermediate file (refer to Section 3 for more detailed description). The intermediate file contains a compacted description of the U/A modeling conception. This file may be retained and made accessible to the U/A using the file manipulation and text editor software of the host computer system. Adjustment and reconfiguration of the U/A's model can thus be accomplished by modifying a previously configured TSM through direct manipulation of the contents of the intermediate file associated with the previous TSM. A particularly adept ICSSM U/A could also conceivably create a model directly in the compact format of the intermediate file (via the host's text editor), bypassing the interactive session required by the model configuration program (see Section 3).

By modifying a previously configured TSM via manipulation of the contents of the relevant intermediate file, the U/A can make adjustments in the TSM, submit the modified TSM specifications for compilation and exercise the modified TSM. This process may be performed iteratively, saving the U/A effort otherwise involved in completely redefining the TSM for each iteration.

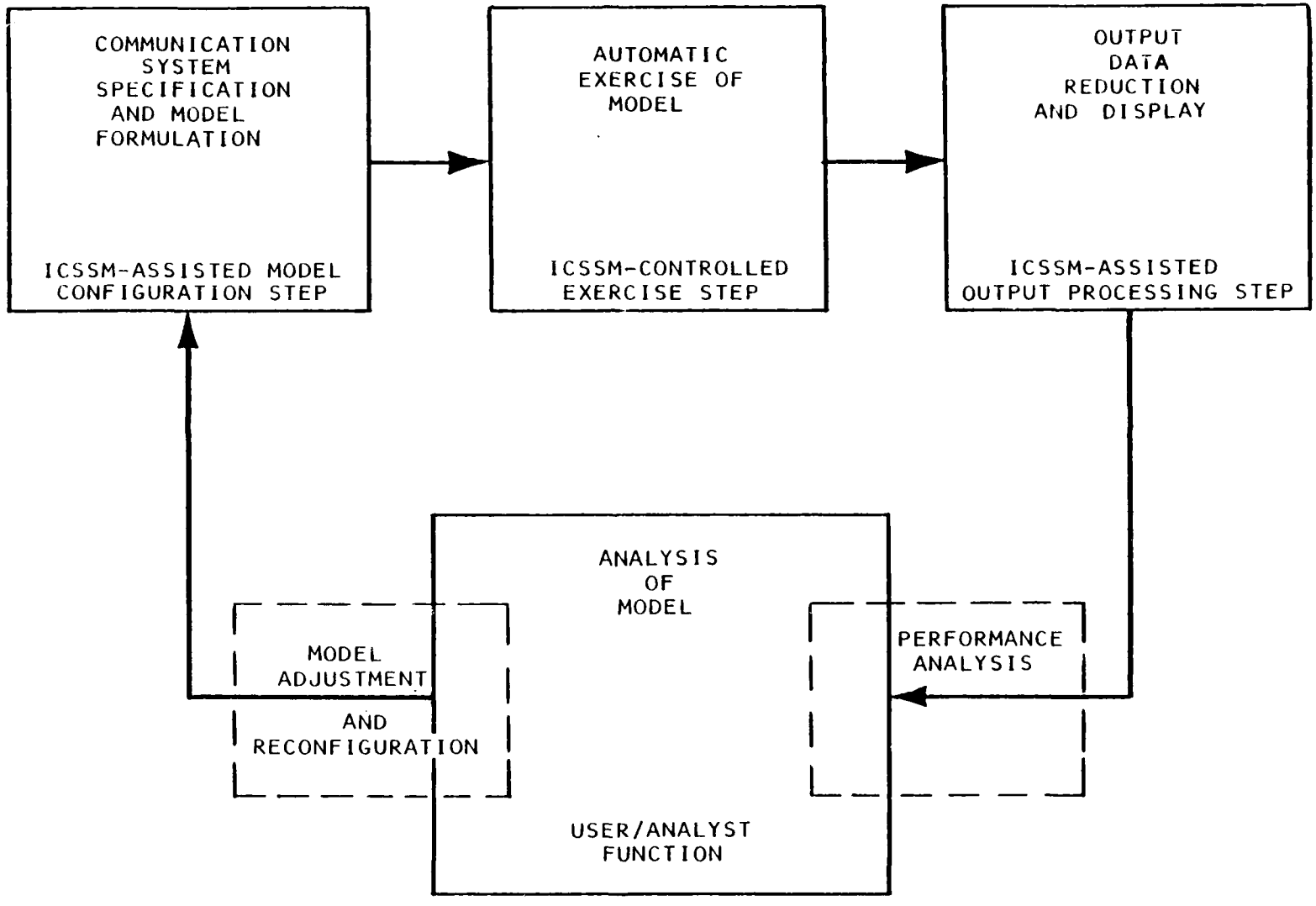


Figure 2-5. ICSSM's Role in Design and Analysis of a Communication System

2.5.3 Partial Model/Experimental Data Use of ICSSM

Circumstances may be that a communications system to be modeled is so "large" or elaborate that it is convenient to "break it up" into segments or partial models. One convenient place to break up a model is at a natural interface (say, at the interface between that portion of the model representing the propagation channel/medium and that representing the receiver antenna). For example, the transmitter/channel/medium portion of a TSM is configured and exercised as an independent TSM (sub-model) under ICSSM control. Exercising this sub-model results in the collection of data simulating the effects of propagation anomalies (eg, multipath); medium disturbances (eg, dispersion); jamming, and noise, on the simulated communication transmission. The data so collected is then formated for use as an input data stream to a second, independent sub-model of the receiver antenna/demod/message transduction portion of the communication system, this portion having its own TSM configured and operating under ICSSM control. The "sizes" of the sub-models (there could be more than the two described in the foregoing example) would be smaller than the entire original TSM and thus more easily matched to the limitations imposed by host computer capacities and by computer operational considerations. Figure 2-6 depicts ICSSM use under the "partial models" method.

The "partial models" method of ICSSM use is particularly relevant to some of the common requirement in communications systems analysis/design. For example, a frequent analysis task addresses the comparison of the performance of several alternative designs under identical conditions. ICSSM, when applied in this context, may be used to construct a sub-model configured to derive realistic transmission/distortion/disturbance data characterizing a given communication channel. The data is collected and applied successively to separate sub-models configured to model the alternative receiver designs. The output data obtained from the

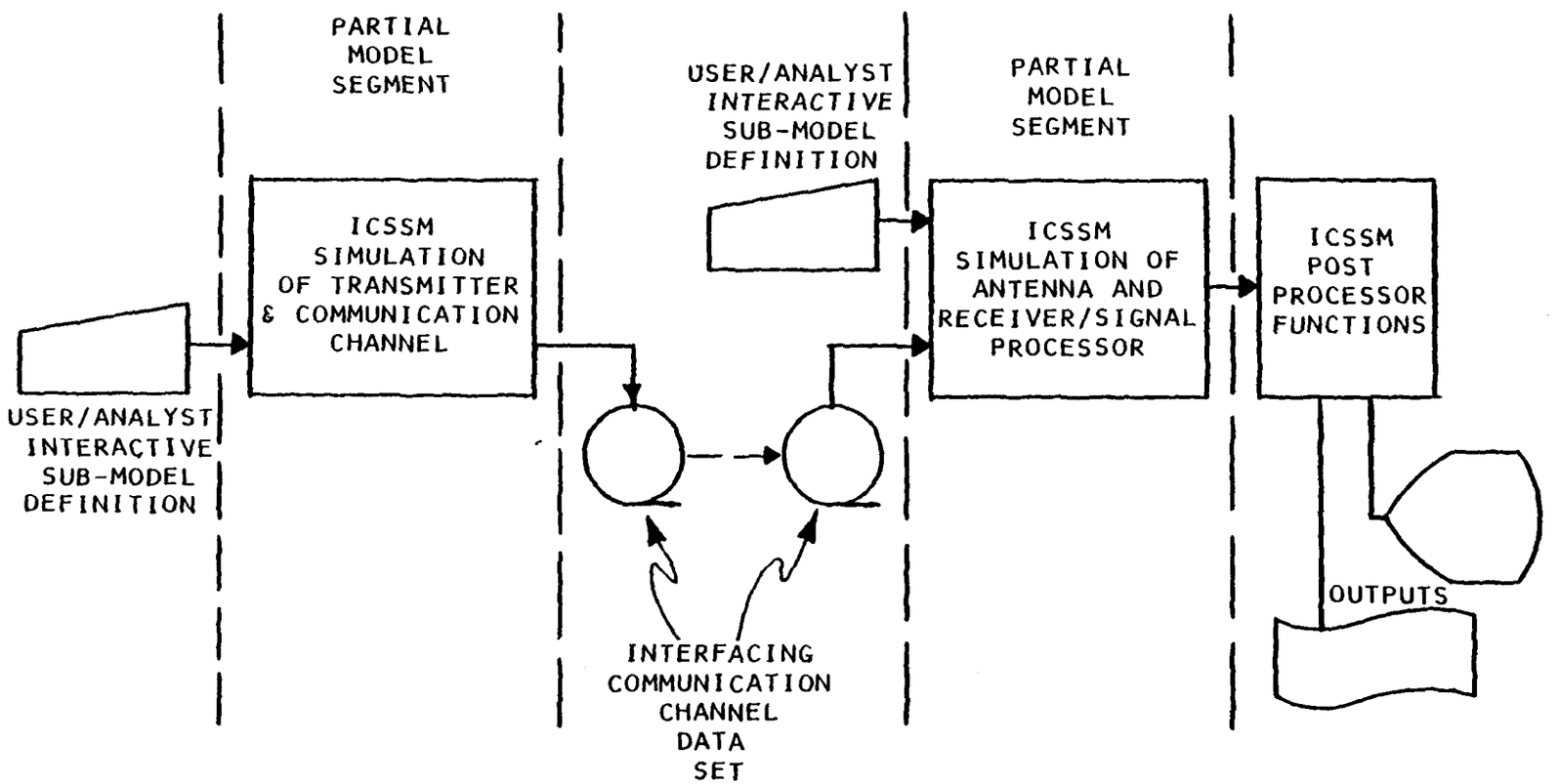


Figure 2-6. ICSSM Usage - An Example of the Partial Models Method

several receiver simulations can be compared directly, since they are obtained under the identical simulated conditions of transmission. Similiarly, experimental data, taken on an actual channel, could be substituted for the simulated channel data.

2.6 ICSSM LIBRARY MODULE RE-ENTRANT DESIGN

The ICSSM TSM incorporates specific functional modules which may be selected by the U/A in the model configuration step. By this means, the general structure of the simulation executive is rendered specific, thereby subsuming the simulation requirements of the particular TSM. Communications-theoretic processing and algorithmic manipulations are performed in "attached" applications modules. The applications modules are re-entrant: data needed as input by a particular module is made available to it through a collection of FORTRAN COMMON areas. Control signals and parameter values required by the modules are similarly provided. Operations on the data are performed by the modules without altering contents of the COMMON blocks (input areas). Results of the module operations are placed in FORTRAN COMMON blocks (output areas) for subsequent use by other elements of the TSM. When all operations of the module have been terminated, all particular data resulting from operations of that module reside in main storage areas --- the module internal structure remains unaltered.

Individual module operation is initiated within the TSM when particularly designated conditions have been established in the control sections of the COMMON area. These conditions are the same for all applications modules. Initiating module operation is coincident with the T_{ON} time-marks of figure 2-7. Termination of module operation is coincident with the pair-wise corresponding T_{OFF}. During module execution, SIMTIME is clocked by the segment-time variable T_{OK}.

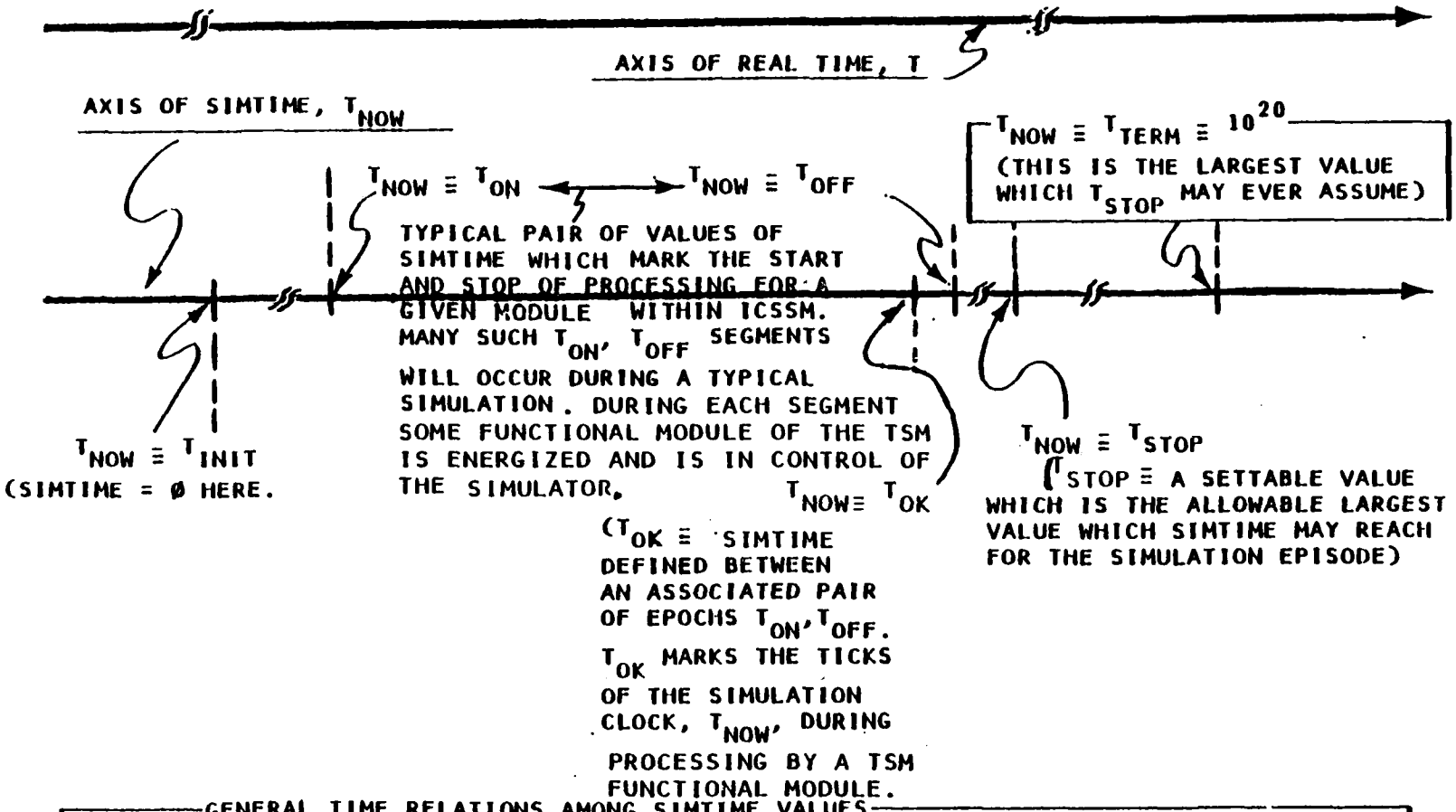


Figure 2-7. SIMTIME (Simulation Time) Relationships Within the ICSSM

By means of comparisons among specific values of T_{ON}, T_{OK}, and T_{OFF}. (say T_{ON} , T_{OKi}, T_{OFFi} compared with T_{ONi+1}, T_{OFFi+1}), concurrent processes manifested in the originating communications system model are converted into sequential processes occurring within the TSM.

2.7 EVENT STEP SIMULATION

ICSSM is an event-driven or event-step simulator. Event-step programming is a useful technique for developing efficient and fast-running computer models. The advantages over pure time-step programs are: (1) events can be scheduled at time intervals appropriate to the processes which they represent; thus, when little activity is occurring in a given module, the events can be widely spaced in time, while for very active modules the events can be closely spaced; (2) events can be scheduled on a real-time basis for some occurances and on a "zero-time" basis when they represent pseudo-events (such as the actions of a bit-error computation module); (3) the Event List can be User-defined and new event types can be added as required; (4) events are controllable in that each process can define an event-time for strobing the system and can schedule events for activating other interconnected system elements.

ICSSM models actually employ both event-step and time-step simulation management:

1. ICSSM employs a simulation clock which measures simulated real-time (SIMTIME) as it transpires during exercises. The diagram of figure 2-7 describes the time relationships which exist during any simulation episode.
2. ICSSM employs an Event Queue, the entries of which either are relatable to explicit processes occurring, or cause explicit processes to occur, during simulation. The diagram of figure 2-8 shows the hierarchical relationships which are established among SIMTIME, events in the Event Queue, and

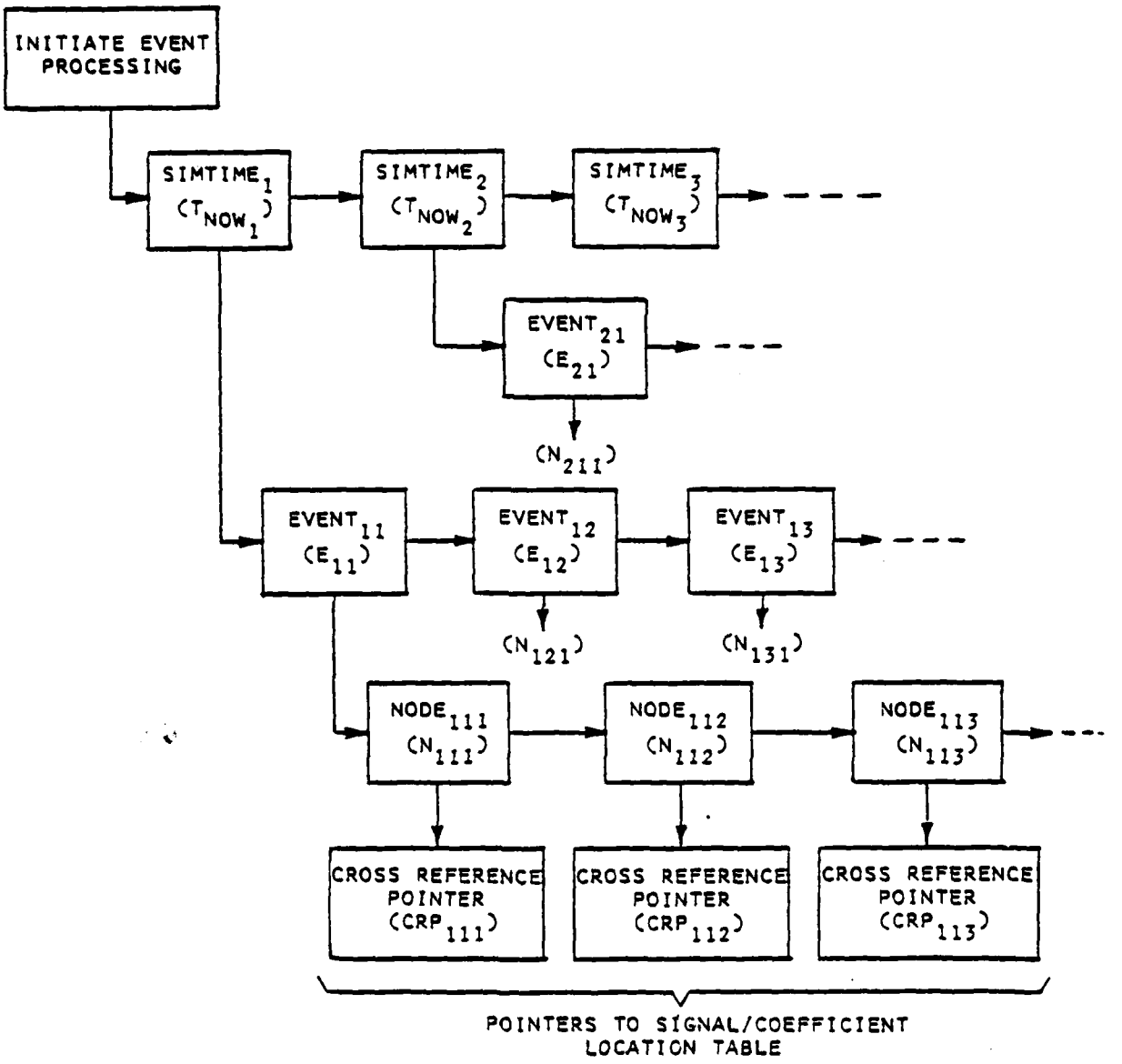


Figure 2-8. Hierarchy of Entries In Event Queue Table

and the "nodes" of the communications system being represented in the TSM.

Section 3

IMPLEMENTATION OF THE INITIAL ICSSM SYSTEM

This Section describes the implementation of the Initial ICSSM system. The description is in three parts:

(1) Paragraph 3.1 et seq provides a general overview of the ICSSM system implementation.

(2) Paragraph 3.2 et seq provides a more detailed discussion of the ICSSM Simulator Component implementation down to the computer program level.

(3) Paragraph 3.3 et seq provides a more detailed discussion of the ICSSM Applications Library Component implementation down to the computer program level.

For more detailed explanation of ICSSM implementation, please refer to the various ICSSM documents (referenced in Appendix A) as required.

3.1 GENERAL DESCRIPTION OF ICSSM SYSTEM IMPLEMENTATION

The Initial ICSSM system-subsystem structure is described in figure 3-1. ICSSM comprises two components: a Simulation Component (SC), and an Application Library Component (ALC).

The SC provides: (1) input and model formulation capability to a prospective user via an interactive i terface; (2) control and housekeeping facilities needed to carry out simulation; and (3) output data reduction/display facilities for recording or examining simulation results.

The ALC contains algorithmic implementations of communication system functional elements, and communications system test equipment or test/measurement methods. These may (through agencies of the SC) be incorporated into communications systems model formulated by a user, be exercised together, and the results recorded.

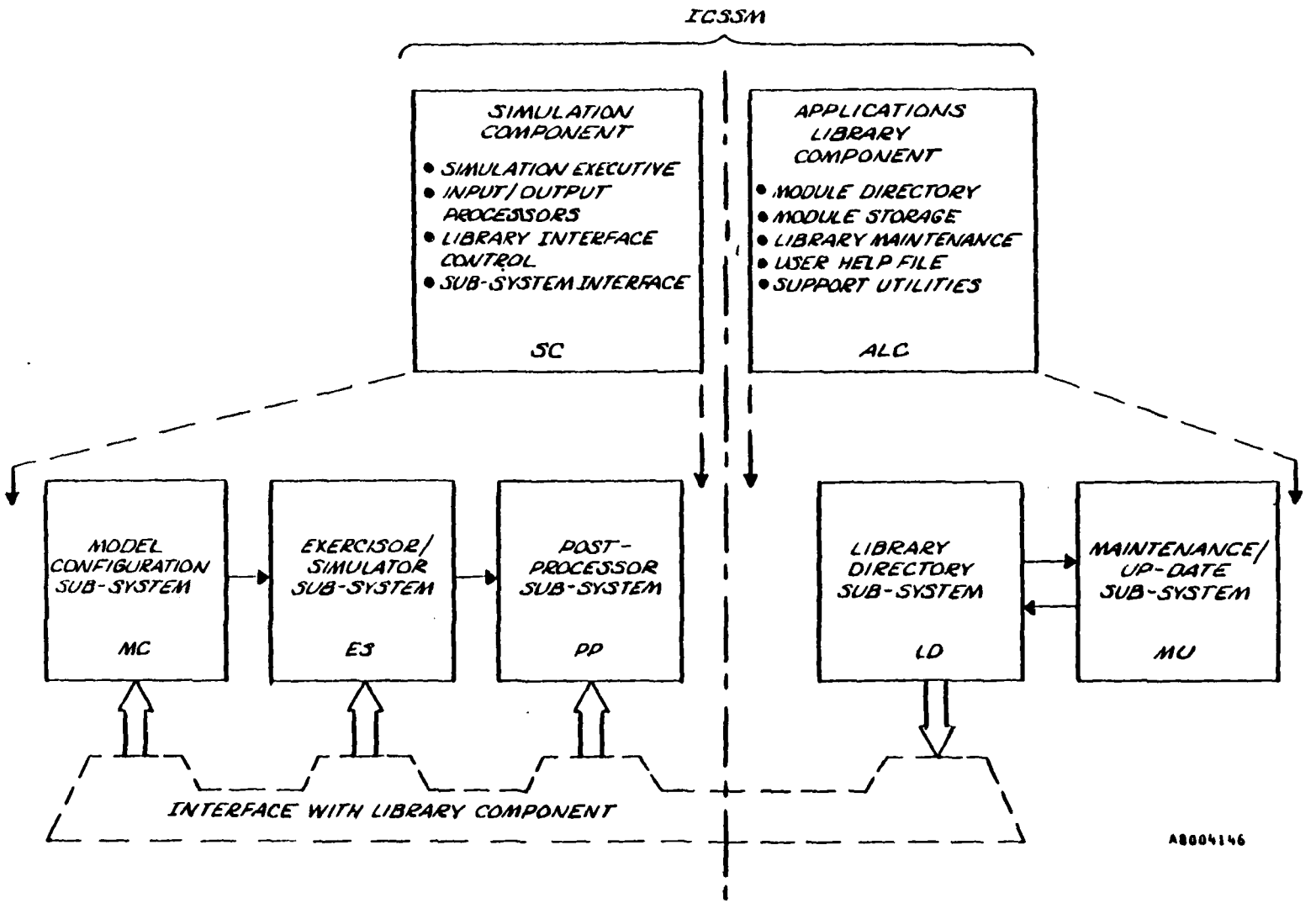


Figure 3-1. System and Subsystem Structure of ICSSM

The diagram of figure 3-2 indicates the relationship between the ICSSM system and the user. The ICSSM system overseas the configuration of a simulation model representing a communications system under study. The ICSSM system also overseas the exercise of the model, and the subsequent data reduction and display of results produced by the simulation. The subsequent paragraphs of the present Section will describe the structure for each of the aforementioned components of ICSSM.

3.1.1 General Organization of the ICSSM Simulation Component

The Simulation Component of ICSSM comprises three subsystems (see figure 3-1):

1. The Model Configurator (MC) subsystem.
2. The Exercisor/Simulator (ES) subsystem.
3. The Post-Processor (PP) subsystem.

The MC subsystem consists of two programs: (1) the Select (MCS) program and (2) the Pre-compiler (MCP) program; and associated computer files.

The MC subsystem provides facilities whereby a user interactively configures a communication system model, the configured-model information being used to produce a simulation model of the communications system of interest. The MC subsystem provides an interactive facility whereby the user via a crt terminal, selects and specifies: (1) functional elements to be included in the model; (2) interconnections among the selected elements; and (3) locations (ports) within the specified model from which output signals are to be drawn for subsequent data reduction and evaluation.

The ES subsystem consists of the Exercisor Kernal (EK), an executive computer program, conjoined with a software version of the simulation model configured via the MC subsystem. The

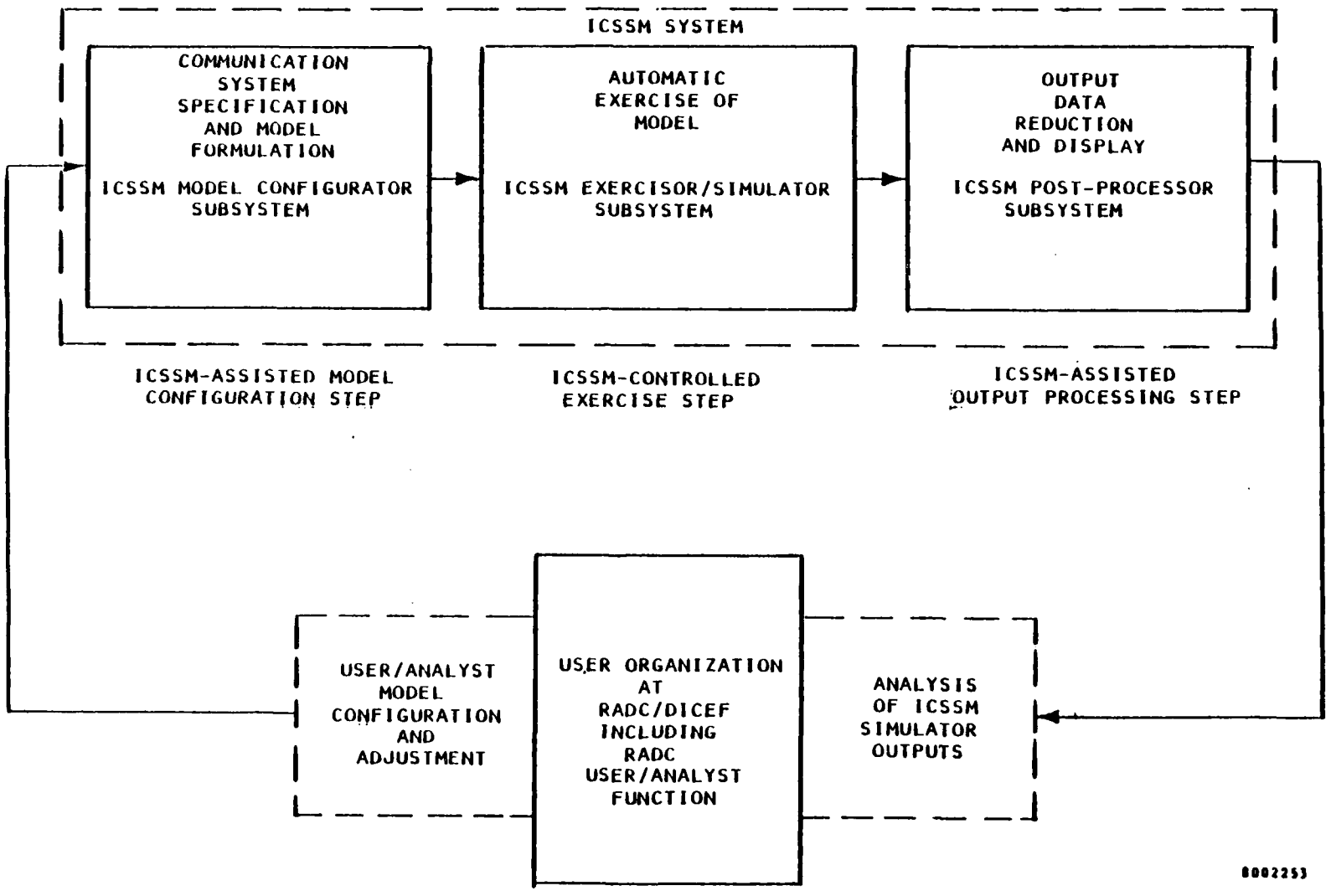


Figure 3-2. Relationships Between ICSSM and User

conjoint program so produced implements the communication system model ("target" simulation model or "TSM") of interest. The TSM may then be exercised under control of the host computer's operating system as would any other application program, producing simulation results (data) which are stored on certain computer files for subsequent analysis.

The PP subsystem consists of two computer programs: the PP Selector (PPS) program; and the PP Exercisor (PPE) program. The data from the simulation executed by the ES subsystem may be submitted to the PP subsystem along with "instructions" contained in a computer file automatically generated by the MC subsystem. These instructions, together with additional data introduced via interactive facilities provided by the PPS program, furnish the means whereby: (1) data reduction/data manipulation processes (eg, Fourier Transformation, Statistics) may be applied to the data resulting from the simulation episode supported in the ES subsystem, and (2) results of these processes may be displayed to the user for his evaluation and subsequent use.

3.1.2 General Organization of the ICSSM Applications Library Component

The Applications Library Component depicted in figure 2-1 comprises two subsystems:

a. A Library/Directory (LD) subsystem

b. A Maintenance/update (MU) subsystem

The LD subsystem is a group of computer files. Some of these files contain subroutines which are algorithmic embodiments of communications system functional processes (eg, matched filter, envelope detector, decoder, propagation channels, antennae), or are utility and support subroutines designed and intended for use in testing or validating ICSSM operation. Other files of the LD subsystem comprise a direc-

tory/index of the functional processes (modules) provided in the library files. The directory and library files collectively supply the algorithmic support for the Simulation Component (SC).

The MU subsystem consists of a computer program which may be used to add functional modeling elements to the files of the LD system.

3.1.3 Subsystem Structures of ICSSM Components

This paragraph expands on the brief descriptions of the subsystems provided above. Both Components of ICSSM are considered, and the relationships among the subsystems are described. The general flow of data and control within ICSSM is depicted in figure 3-3.

3.1.3.1 General Description of the ICSSM Simulation Component

Figure 3-4 shows that the MC subsystem consists of two computer programs - the Select (MCS) program and the Pre-compiler (MCP) program - and certain related files. Figure 3-5 shows that the ES subsystem consists of the EK program and the Process Module (PM). The PM is a specially prepared, model-dependent, and automatically written FORTRAN subroutine which contains information describing the communication system model that was entered by the user in interactive operation of the MC subsystem. The PP subsystem consists of two programs: the PPS program and the PPE program; and certain related files, as shown in figure 3-6.

3.1.3.1.1 General Description of the Model Configuration Select (MCS) Program

The MCS program operates interactively to:

- Aid the user in reviewing the contents of the Library to locate suitable modeling elements for the simulation application of interest.

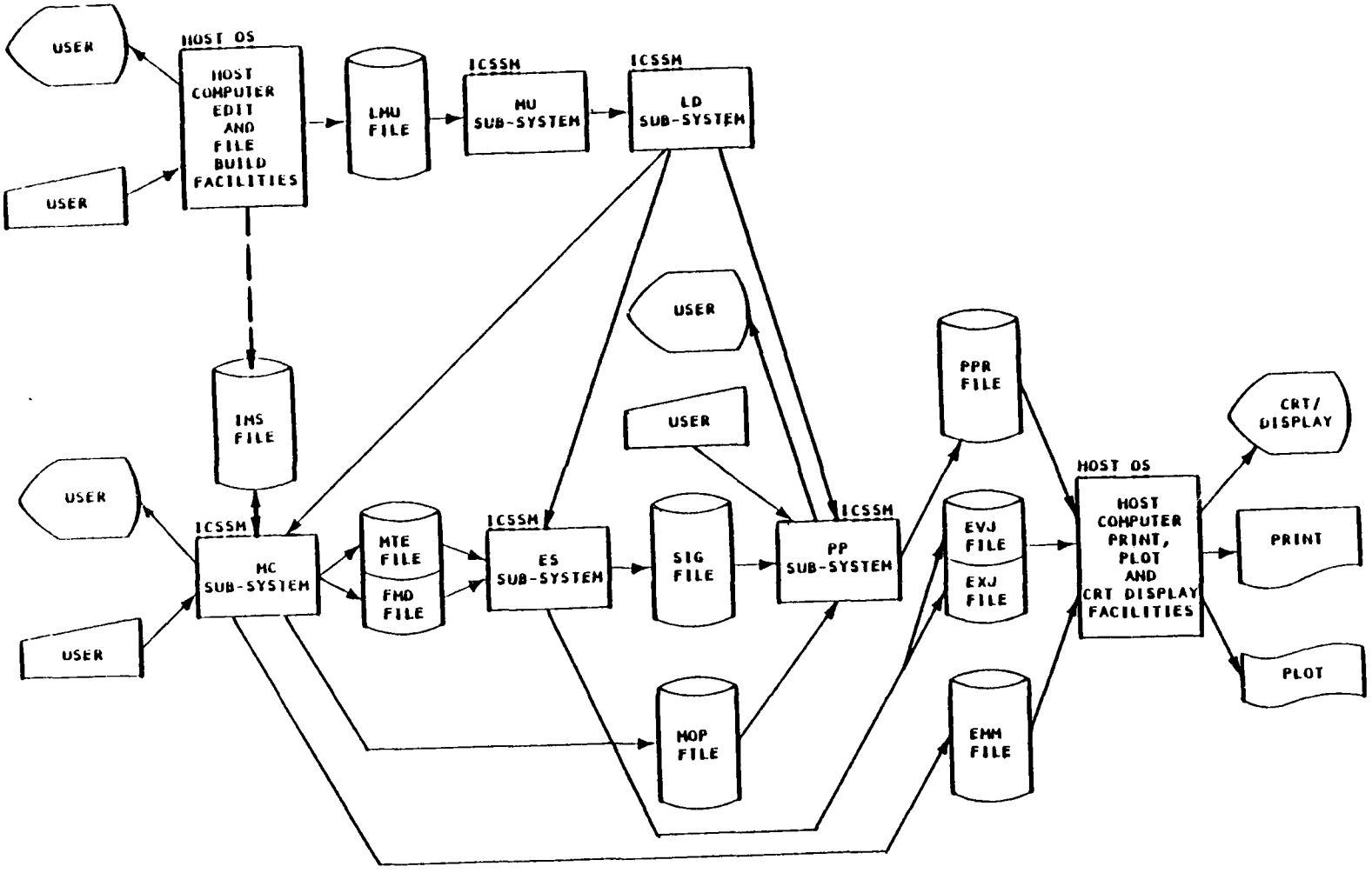


Figure 3-3. General Flow for ICSSM System

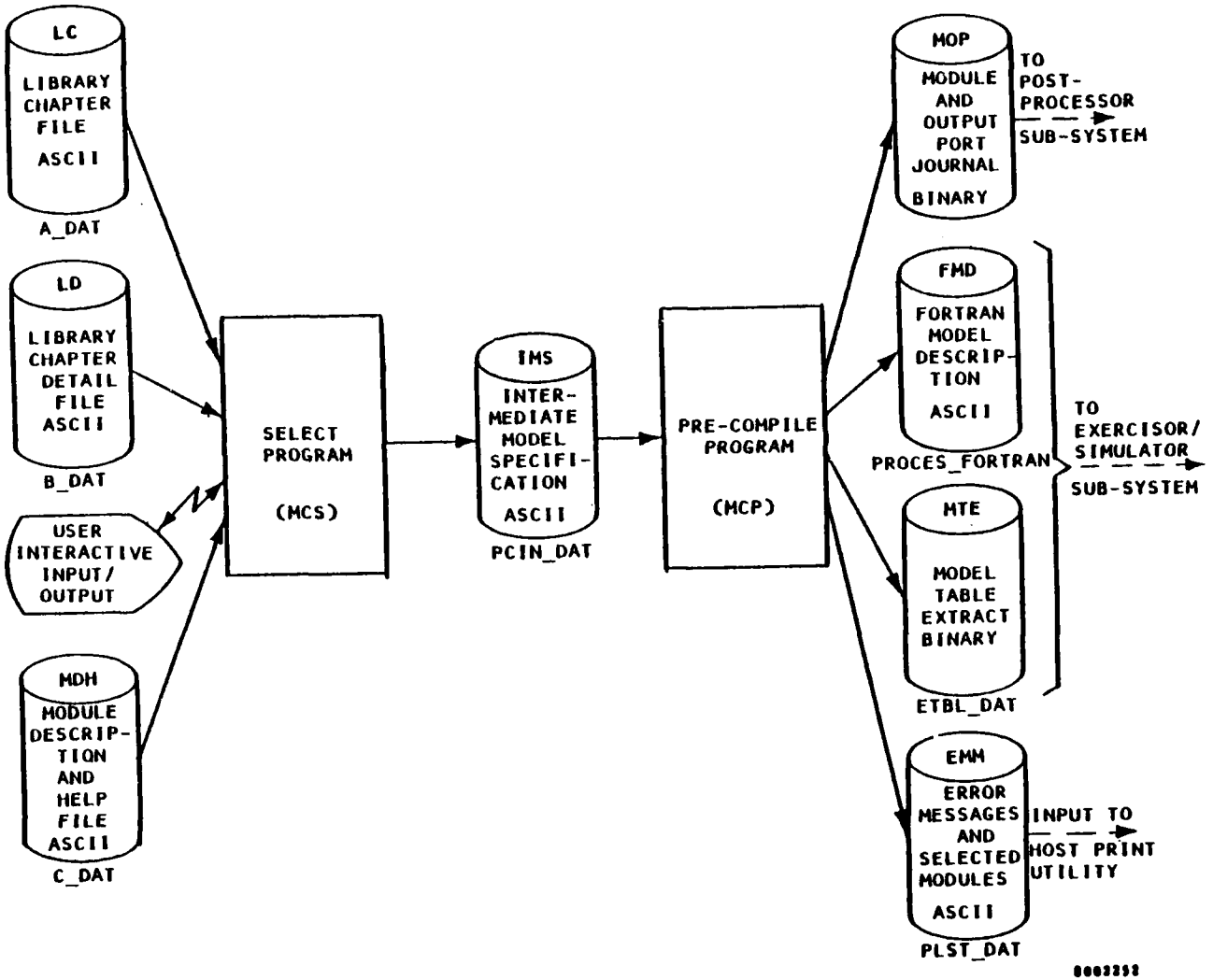


Figure 3-4 Structure of Model Configurator (MC) Subsystem

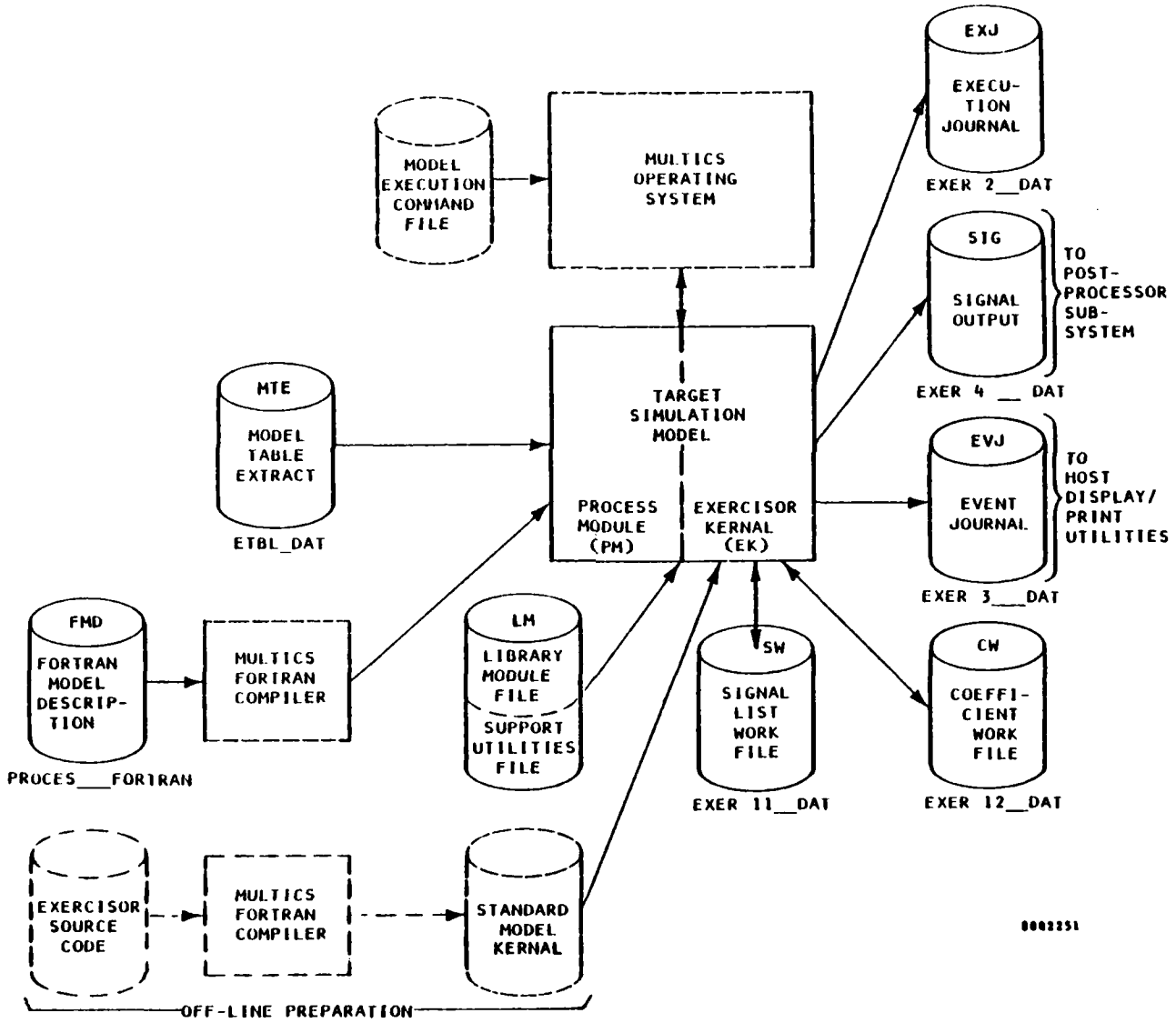


Figure 3-5. Structure of Exercisor/Simulator (ES) Subsystem

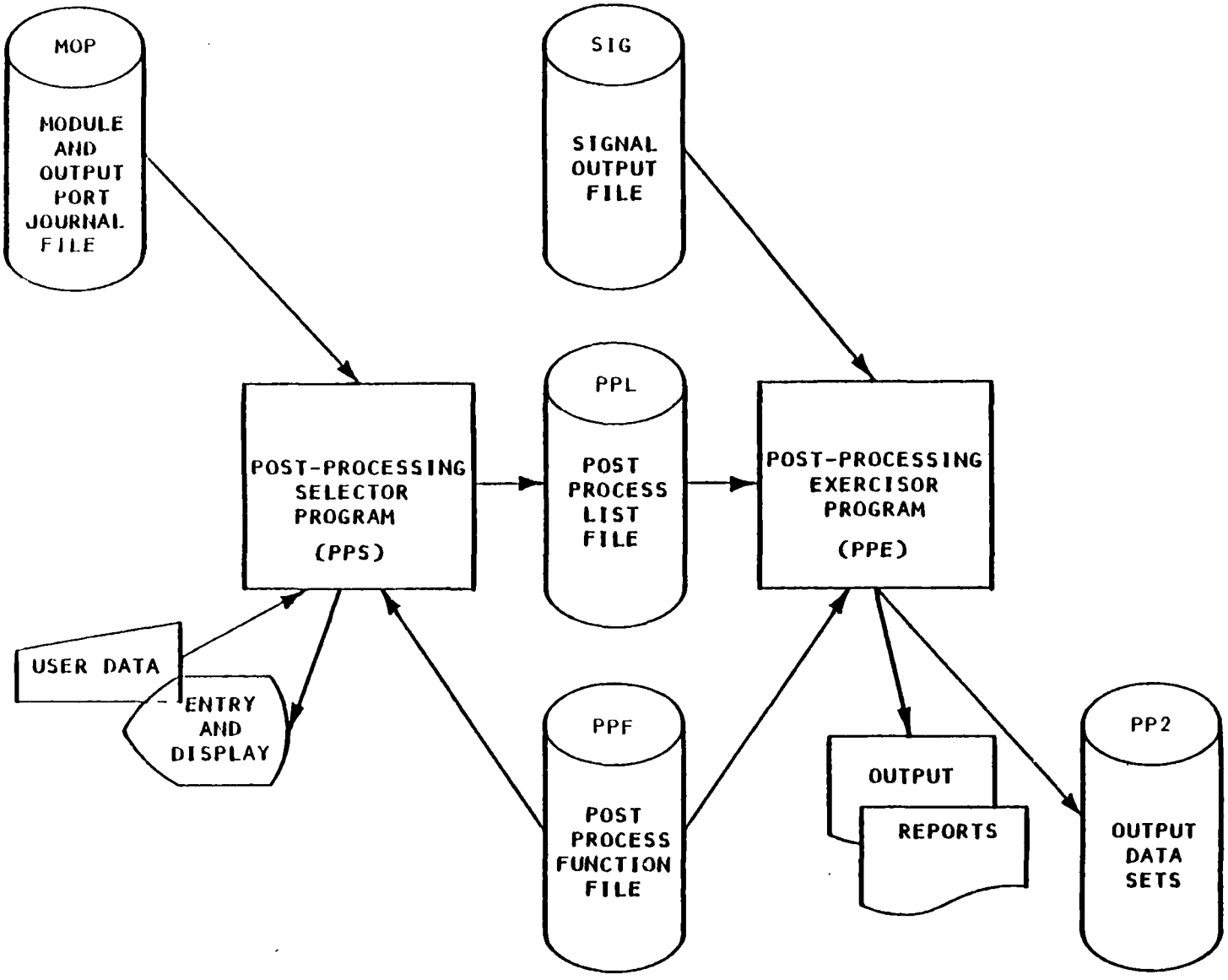


Figure 3-6. Structure of Post-Processor (PP) Subsystem

- o Prompt the user to provide required inter-element connections which reflect the simulation application of interest.
- o Provide means for accepting user-specified values for settable parameters which may be associated with modeling elements selected.

The MCS program cperates in two steps. The first step provides interactive facilities for the selection of Applications Library elements (viz, modules) and for the registration of the values of settable parameters associated with those modules selected.

The second step, entered only after step one is completed, provides interactive facilities for specifying module interconnections.

Upon initial interaction with the MCS program, the user sees a "menu" displayed on the crt terminal providing four choices for use of the MCS program. Termination of any subsequent interactive operation returns the user to this display/select menu.

The MCS program performs two kinds of validation of model data input:

a. For each module selected, the MCS program requires the specification of connections between each and every available module output port and some input port of some module. Attempts to terminate the declaration of connection data concerning a given module, before connection specifications for all output ports of that module have been inserted, cause an appropriate diagnostic to appear on the crt terminal. The interactive sequence for specification of connections "from" the module in question is reinitiated.

b. For each module specified in the model, the MCS program accumulates the connection data specified for each module and scans the data looking for unconnected input ports. For each unconnected input port an appropriate diagnostic is issued to the crt terminal requiring the user to reestablish connection data for some module (of his own choosing) which expunges the open-port condition detected. The user is not permitted at this point in model configuration, to add a new module to the model, since the assumption is that the user specifies all modules of his model in step 1. Thus, the only remaining possibility is that the user failed to completely specify a "fan-out" connection (ie, two or more connections from the same output port of a module) intended for some module in his model.

For each reassertion of output connections for some module, the completeness of output port connections is re-checked for open input ports, as described above. The MCS program continues the interactive validation sequence until both conditions being validated are fully satisfied.

At the successful completion of the connection validation process, MCS program operation terminates.

3.1.3.1.2 General Description of Model Configuration Pre-Compiler Program (MCP)

The MCP program accepts output of the MCS program and produces the FORTRAN Model Description (FMD) file. The FMD file contains a model-dependent version of the Process Module, the application-specific modeling segment which joins with the Exercisor Kernal to make up an individual, specific "target simulation model" (TSM).

3.1.3.1.3 General Description of the Exercisor Simulator Subsystem/Target Simulation Model (TSM) Program

Each TSM program consists of a fixed or programatically constant simulation executive (Exercisor Kernal) and a customized simulation model segment (Process Module) automatically constructed by prior operation of the MCS and MCP programs. These are linked to form a complete customized simulator, known as the TSM. Each modeling conception rendered to the ICSSM system, via the interactive MCS program, gives rise to a unique version of the TSM. Exercise of the resultant TSM produces simulation output data reflecting the behavior of simulated communication signals appearing at each connection node of the communication system model. Output data is stored in the ICSSM system's Signal Output file for subsequent analysis. The TSM is the center of simulation activity in the ICSSM system.

Principle elements which comprise the Exercisor Simulator (ES) subsystem are shown in figure 3-5, along with ancillary computer system elements. The FMD file produced by the MC subsystem is submitted to the ICSSM host computer's FORTRAN compiler. By this mechanism, the model-specific Process Module (PM) segment is created for use with the Exercisor Kernal (EK). The EK (FORTRAN-coded and submitted previously to the FORTRAN compiler) is available in an executable form, invariant from TSM to TSM.

3.1.3.1.4 General Description of the Post-Processor Selector Program

The principle elements of the Post-Processing subsystem are depicted in figure 3-6.

The Post-Processor Selector (PPS) program provides interactive crt-terminal-oriented access to the ICSSM system to:

- Aid the user in selecting post-processing functions, algorithms, and operations.
- Prompt the user to designate to which TSM signal outputs the post-processing functions are to be applied.
- Accept user specified values for setable parameters associated with the post-processing function(s) selected.

In summary, the PPS program is used to specify the output data reduction and algorithmic processing to be applied to data generated by execution of an ICSSM TSM.

The PPS program interacts with the user, via a crt terminal, in a prompt/response mode:

- To retrieve and display tutorial information describing the capabilities and use of the ICSSM post-processing facilities.
- To retrieve and display information on algorithmic processes which might be applied to signals generated within the TSM.
- To retrieve and display information concerning the origin and nature of the TSM output signals that are accessible for post-processing.

Based on user responses, the PPS automatically generates all instructions and prescriptions needed in carrying out the data reduction and processing designated.

3.1.3.1.5 General Description of the Post-Processor Exercisor Program

The Post-Processor Exercisor (PPE) program accepts instructions and execution parameters produced by PPS program execution, and processes TSM-generated signals accordingly. The signals to be processed are available to it in the Signal Output (SIG) file. The results of PPE execution are available for subsequent display and examination.

3.1.3.2 General Description of the ICSSM Applications Library Component

The subsystem organization of figure 3-1 indicates that the Applications Library Component (ALC) is composed of two subsystems: the LD subsystem (figure 3-7), and the MU subsystem (figure 3-8).

The LD subsystem comprises six data sets organized into an Applications group and a Utilities group. These files are described in paragraphs 3.1.3.2.1 and 3.1.3.2.2.

The MU subsystem comprises a single program --- the Library Maintenance/Update (LMU) program. This program is described in paragraph 3.2.7.

3.1.3.2.1 General Description of the Library/Directory Applications Group (LDAG)

The LDAG provides on-line storage for the application modules (functional elements required for communications system modeling) which reside in the LM files. This file contains all subroutines which may be employed in configuring the TSM.

The LM file contents are divided conceptually into "Chapters", based upon the taxonomy employed in classifying the modules contained in the LD. The Chapter (LC) file contains data on the most general classification of application modules.

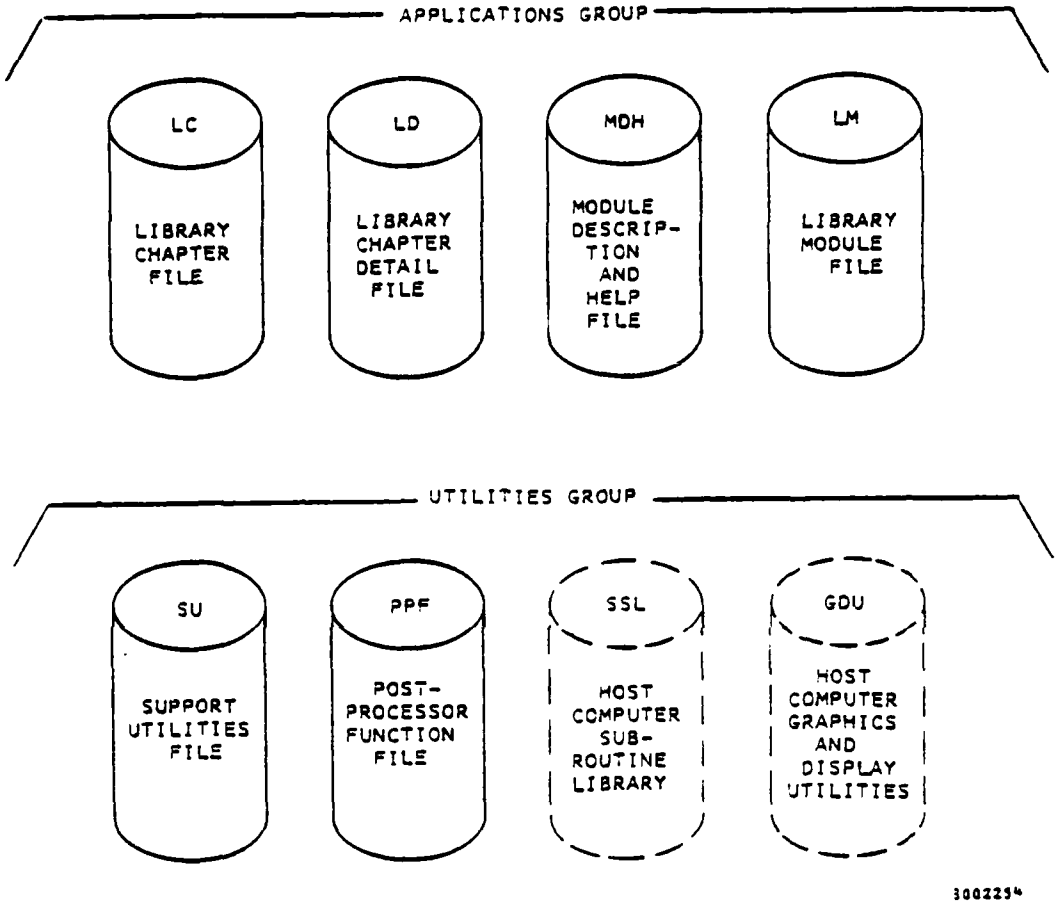


Figure 3-7. Structure of Library/Directory (LD) Subsystem

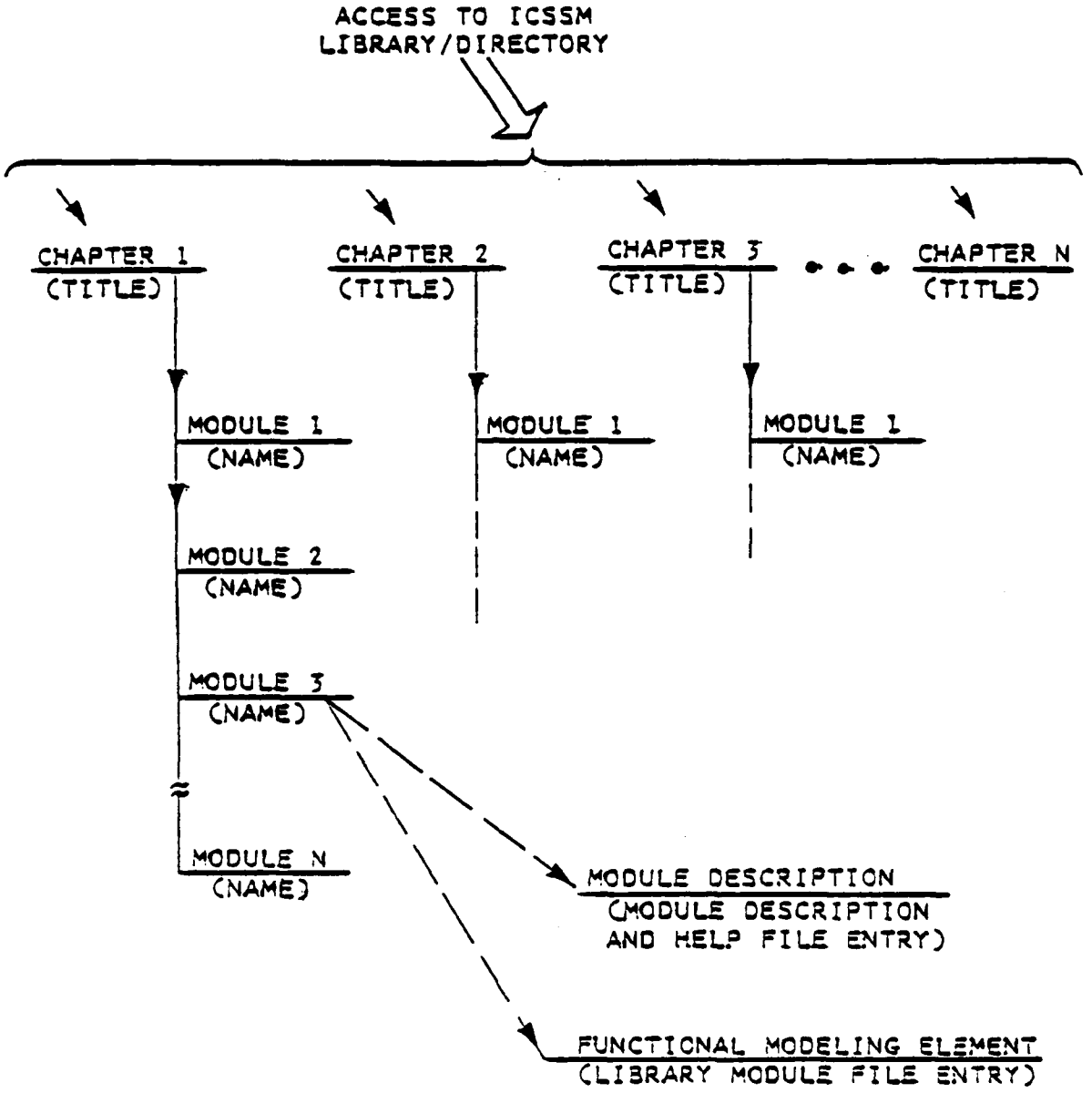


Figure 3-8. Hierarchic Organization of ICSSM Library Directory Applications Group (LDAG)

The Chapter Detail (LD) file contains data describing those application modules assigned to the chapters delineated in the Chapter file. The structure of these two files is analogous to chapter headings and within-chapter details provided by the "Table of Contents" of most textbooks.

The Module Description and Help (MDH) file contains an entry for each application module registered in the LM file. The user accesses the MDH file using the facilities of the MCS program.

The hierarchic structure of the LDAG is described in figure 3-8.

The LM file stores FORTRAN subroutines employed in constructing TSM. Two classes of subroutines exist in the LM file: Class 1 subroutines are cataloged in the LD file and are employable directly as model elements in configuring the TSM.

Class 2 subroutines are not cataloged in the LD file. Class 2 subroutines are dependent subroutines employed by the subroutines in Class 1, but not directly as modeling elements.

Class 1 subroutines are referred to as "modules." Class 2 subroutines are referred to as "dependent modeling subroutines" (DMS).

3.1.3.2.2 General Description of the Library/Directory Utilities Group (LDUG)

The LDUG stores subroutines and other program elements required by the programs which constitute the ICSSM system. The LDUG comprises the Support Utilities (SU) file and the Post-Processor Function (PPF) file.

SU file entries are subroutines which may be employed by programs of the ICSSM system as required (eg, internal to the ICSSM executive software or internal to the modules residing in the LM file). The SU file entries are not employed directly as communications modeling elements but perform common data-manipulative or control functions and services required in the programs of the ICSSM system.

The PPF file entries are employed automatically by the PPE program in data reduction and signal processing operations performed upon signals derived from the TSM, as designated in normal operation of the ICSSM facilities.

NOTE

Figure 3-7 indicates two additional "files" within the LDUG. These are: (1) The host computer scientific subroutine library (SSL), and (2) The host computer graphics and display utilities (GDU). While these "files" do not form a part of the ICSSM software per se, they are conceptually a part of the total processing capability employable by the ICSSM user.

3.2 DETAILS OF ICSSM SIMULATION COMPONENT IMPLEMENTATION

The following paragraphs provide descriptions of the programs subroutines and files comprising the ICSSM Simulator Component. The programs described are:

- o Model Configurator Select (MCS) Program
- o Model Configurator Pre-compiler (MCP) Program
- o Exercisor/Simulator Program (the Kernal of each TSM)
- o Post-Processing Selector (PPS) Program
- o Post-Processing Exercisor (PPE) Program

3.2.1 Description of the MC Select (MCS) Program

The MCS program provides terminal-oriented access to the ICSSM system.

Execution of the selection and interconnection phases of the MCS program results in the generation of the Intermediate Model Specification (IMS) file.

The IMS file contains an abstracted, compacted description of the specified model and contains all the module identifiers, parameter values, and interconnection information needed to define the simulation to be performed. The IMS file is the coupling vehicle to the MCP program. It contains all the data needed to produce the FORTRAN Model Description (FMD) file. The FMD file is the output product of the MCP program execution.

NOTE

When employed with the MULTICS system, the MCS program also generates the FCBA file, containing automatically composed MULTICS SFC commands. This is produced as a subsidiary output of the MCS program. The FCBA file is generated in compliance with the

requirement (when executing FORTRAN programs under the MULTICS operation system) to perform in-main-storage alignment of FORTRAN COMMON blocks among a FORTRAN mainline and its CALLed subroutines. The SFC commands required differ from TSM to TSM, as different application modules are expected to be selected for use in different TSM.

The MCS program uses as input files:

(1) The Library Chapter (LC) file

(2) The Library Chapter Detail (LD) file

(3) The Module Description and Help (MDH) file

The data from these files are employed during the interactive TSM Module-selection/Model-configuration session afforded by the MCS program.

The MCS program implements model validation (ie, consistency and completeness) checks on user-input data (refer to paragraph 3.1.3.1.1 and table 3-1).

At the successful completion of the validation process, IMS and FCBA files are written, thus ensuring completely validated information for further processing. MCS program operation then terminates.

Program termination in the MCS program is automatic, requiring no overt action of the User, except in following the prompts presented to him.

Table 3-1. Error Checking/Data Validation in the MCS Program

Data Item Checked	Check/Validation Applied
o Module name	Validity of the module name entered.
o Module parameters	Validity of the form and size of parameter values entered. (No range-limit or reasonableness checking is provided).
o Module connections	Each output port on each module has been connected to an input port on some module.
o Module connections	Each input port on each module is connected to some output port of some module.
o Module connections	The number of input ports (the "fan out") to which each output port is connected does not exceed 5.

3.2.1.1 Major Operations Performed in the MCS Program

The MCS program is organized according to the hierarchy of figure 3-9. The macro-level flowchart of figure 3-10 reflects the two phases in MCS program operation: the first (or select) phase, in which selection of library modules and associated parameter values is performed in interaction with the user; and the second (or interconnection) phase, in which intra-model connection data is entered and model topology is constructed and validated in interaction with the user.

The MCS program control section (root):

- controls switching among program phases and segments
- accepts crt terminal input from, and routes crt display output to, the Terminal I/O Segment.
- provides a menu/display on the crt terminal for interaction with the user.

1. MAIN - MAINLINE FOR MCS PROGRAM

1.1 INITT---INITIALIZE THE TERMINAL AND THE TERMINAL STATUS AREA

1.2 TERM---SPECIFY TERMINAL AS A TEKTRONIX 4014/15 TERMINAL WITH EXTRA FEATURES

1.3 CHRSIZ---PROVIDE CURRENT CHARACTER HEIGHT AND WIDTH IN RASTER UNITS

1.4 ANMODE---INSURE THAT THE TERMINAL IS IN A/N MODE

1.5 OPFIL1---OPEN FILES LC, LD, MDH, IMS, FCBA

1.6 DISPL---DISPLAY SCREEN FOR USER'S CHOICE OF MCS PROGRAM MODE

1.6.1 ERASE---TERMINAL SCREEN ERASED WITHOUT CHANGING THE MODE OR BEAM POSITION

1.6.2 HOME--- MOVE ALPHANUMERIC CURSOR TO UPPER LEFT CORNER OF THE SCREEN

1.6.3 ANMODE---DUMP OUTPUT BUFFER

1.6.4 CHAP---READ IN (LC FILE) LIST OF CHAPTERS FROM LIBRARY

1.6.5 MODUL---LIST MODULES (LD FILE) FOR CURRENT CHAPTER; USER CHOOSES TSM MODULES

1.6.5.1 PARM---USER CHOOSES PARAM VALUES

1.6.6 MODL---LIST ALL CHOSEN MODULES ON CRT FOR USER INSPECTION

Figure 3-9. MCS Program Hierarchy (Sheet 1 of 3)

1.7 PORTD---PROVIDE DISPLAY OF SELECTED TSM MODULES WITH THEIR INPUT AND OUTPUT PORT DESCRIPTORS

1.7.1 HDCOPY---GENERATE A HARDCOPY OF SCREEN CONTENTS

1.7.2 ERASE--- TERMINAL SCREEN ERASE

1.7.3 HOME---MOVE CURSOR TO UPPER LEFT CORNER OF SCREEN

1.7.4 ANMODE---DUMP OUTPUT BUFFER

1.8 INCON---DELEGATE THE INTERCONNECTION OF SELECTED TSM MODULES

1.8.1 HDCOPY---GENERATE A HARDCOPY OF SCREEN CONTENTS

1.8.2 ERASE---TERMINAL SCREEN ERASE

1.8.3 HOME---MOVE CURSOR TO UPPER LEFT CORNER OF SCREEN

1.8.4 ANMODE---DUMP OUTPUT BUFFER

1.8.5 ICHEC--- INITIALIZE THE CHECK-WORD

1.8.5.1 NSET---INITIALIZE THE CHECK WORD

1.8.6 DEC1---READ AND DECODE OUTPUT PORT CONNECTIONS

1.8.6.1 XREAD---READ-IN OUTPUT PORTS AND CONNECTIONS DATA

1.8.6.2 NUMB---DECODE NUMBERS OF CONNECTIONS AND PERFORM ERROR CHECKING

1.8.6.2.1 XREAD

1.8.6.3 NDYN---CHECK THAT THE INPUT PORT HASN'T BEEN CONNECTED YET

Figure 3-9. MCS Program Hierarchy (Sheet 2 of 3)

1.8.7 CHECW---SEARCH CHECKWORDS TO SEE IF ANY INPUT PORTS ARE NOT USED AND ALLOW CONNECTIONS TO UNUSED INPUT PORTS

1.8.7.1 NCHK---CHECKWORD IS CHECKED TO SEE THAT ALL INPUT PORTS HAVE BEEN USED

1.8.7.2 DEC1---READ AND DECODE OUTPUT PORT CONNECTIONS

1.8.7.2.1 XREAD---READ-IN OUTPUT PORTS AND CONNECTIONS DATA

1.8.7.2.2 NUMB---DECODE NUMBERS OF CONNECTIONS AND PERFORM ERROR CHECKING

1.8.7.2.2.1 XREAD

1.8.7.2.3 NDYN---CHECK THAT THE INPUT PORT HASN'T BEEN CONNECTED YET

1.9 FINITT---TERMINATE PROGRAM. OUTPUT CONTENTS OF BUFFER

Figure 3-9. MCS Program Hierarchy (Sheet 3 of 3)

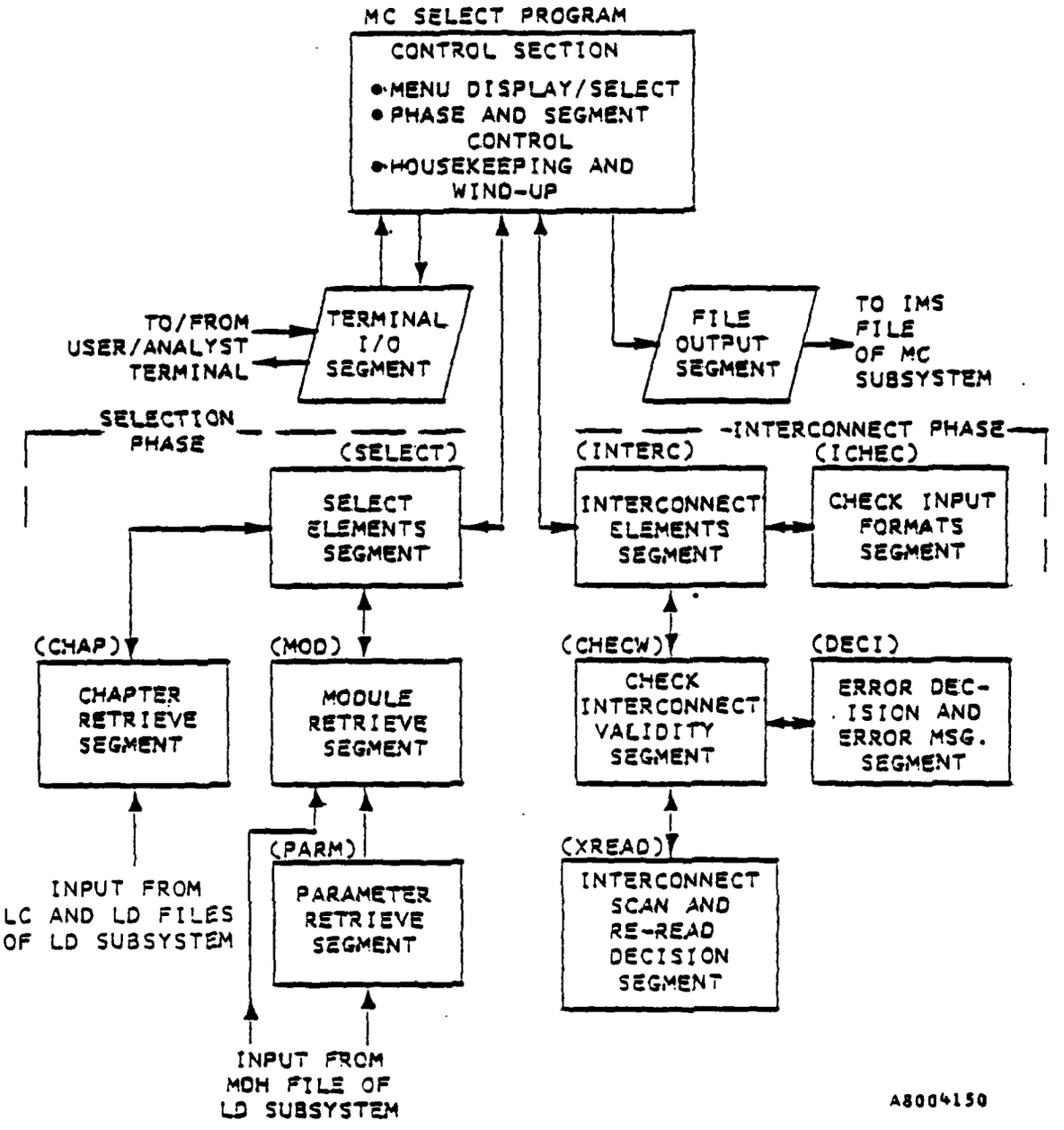


Figure 3-10. Macro-Level Flow Chart of MC Select (MCS) Program

- o initializes program variables, performs general housekeeping functions, and provides orderly conclusion of program operation.

Upon initiation of MCS program execution, a menu display on the user crt provides choice:

#1. for displaying a list of modules in the presently chosen Chapter of the LD file

#2. for displaying a list of Chapters currently provided in the LD file

#3. for displaying a list of modules named for use thus far in the current interactive session

#4. for displaying a prompt/query as to whether the user is ready to begin the interconnecting process

In display #1, module selection is enabled. A list of the modules available in the current Chapter is displayed. The user is prompted to choose a module and provide a "user name" for it, one module at a time. Information associated with the chosen module is displayed before the program accepts another choice or action. Parameter values for the chosen module are accepted by the MCS program at this point.

In display #2, Chapter selection is enabled. The MCS program displays a list of the Chapters currently in the Library. The user is prompted to enter the number of the Chapter desired, one Chapter choice at a time. Module selection from that Chapter is enabled automatically (Display #1) before another Chapter may be chosen.

In display #3, the MCS program displays a list of the modules chosen so far in the session, together with their module numbers, Library-assigned names, user-assigned names, number of parameters, number of input ports, and number of output ports.

In display #4, the MCS program prompts the user to begin the interconnection process.

The interconnect process proceeds with MCS program review of internal table entries pertaining to the selected modules. The MCS program displays each module (table entry) in the order of original encounter in the select phase. The display contains:

- o The Library-registered module name.
- o The module name (ie, user name) assigned by the user during the select phase.

Each output port (number) associated with a table entry is displayed in turn, and the user directed to enter one or more module-number/input-port-number pairs representing desired connections from the output port displayed to input ports of other modules in the set of selected modules. After the user has assigned all desired to connections associated with the module displayed, the next module (table entry) is displayed. This process repeats until the table of selected module entries has been exhausted. The MCS program reviews the connection information to determine if:

a. each output port is connected to at least one input port of some module.

b. each input port on each module is specified as having been connected to one output port of some module.

For each failure of either condition above, the MCS program displays an appropriately worded error message (refer to table 3-1) and all currently registered information relating to the module(s) associated with the connection fault detected. The user is prompted to enter connection information which either redefines the previously inserted offending connection information or adds to it to complete the required interconnection topology. After each such reconnection action has been entered, the table is rescanned in search of then-existing connection faults. This process repeats until all connection faults have been corrected. The interconnect

phase does not permit connection-faults to be corrected by changes to the original list of modules selected during select-phase interaction. Thus, no new modules are accepted during interconnect-phase operation. After all connection faults have been corrected, the MCS program forms the module and connection information into records appropriately formated, and writes these records to the Intermediate Model Specifications (IMS) file.

3.2.2 Description of the MC Pre-Compiler (MCP) Program

The MCP program accepts the contents of the IMS file as input, and produces four files as outputs:

a. The FORTRAN Model Description (FMD) file.

b. The Model Table Extract (MTE) file.

c. The Module and Output Port (MOP) Journal file.

d. The Error Message and Module (EMM) Listing file.

The IMS file is read sequentially, and information destined for the four output files extracted from the records thus read through the internal processing of the MCP program.

The FMD file contains a model-dependent version of the Process Module (PM). Each communication system model submitted to ICSSM for simulation gives rise to a specific, automatically generated version of the PM which, after submission to the FORTRAN Compiler, is included in the TSM.

The MTE file contains tables that record module interconnection data and control data used to regulate and direct the TSM during the course of simulator execution. The tables consist of:

a. The Master Module List: a list of modules named for use in the TSM, each associated with pertinent parameter and control data.

b. The Node List: a list of connection nodes which exist in the TSM.

c. The Parameter List: a list of parameter values associated with each of the modules comprising the TSM.

d. The To-List: a list of module interconnection data which specifies the TSM topology.

The MOP file contains a table of modules to be employed in the TSM along with associated topological parameters of the TSM. The MOP file is employed as an input to the PPS program (refer to paragraph 3.2.4).

The EMM file contains a table summarizing the modules chosen for use in the TSM, the parameters (and assigned values thereof) associated with each module, and the applicable output connections. It also contains diagnostic/error messages generated by the MCP program which reflect error conditions of various kinds encountered during MCP program execution.

Program termination in the MCP program is automatic. However, upon normal termination of the MCP program, the contents of the EMM file may be examined to be sure there are no diagnostic/error messages present there. If there are, the conditions which gave rise to these errors must be corrected and the MCP program executed again. The error conditions and messages in the EMM file are identical to those indicated in table 3-1.

3.2.2.1 Major Operations Performed in the MC Pre-Compiler (MCP) Program

A macro-level flowchart depicting the functional operation for the MCP program is provided in figure 3-11.

The MCP program operates with a two-pass structure: Pass A reads all records offered as input on the IMS file, checks the record syntax and topological consistency of module connection information found there, and builds internal tables required in the formation of the MTE and MOP files.

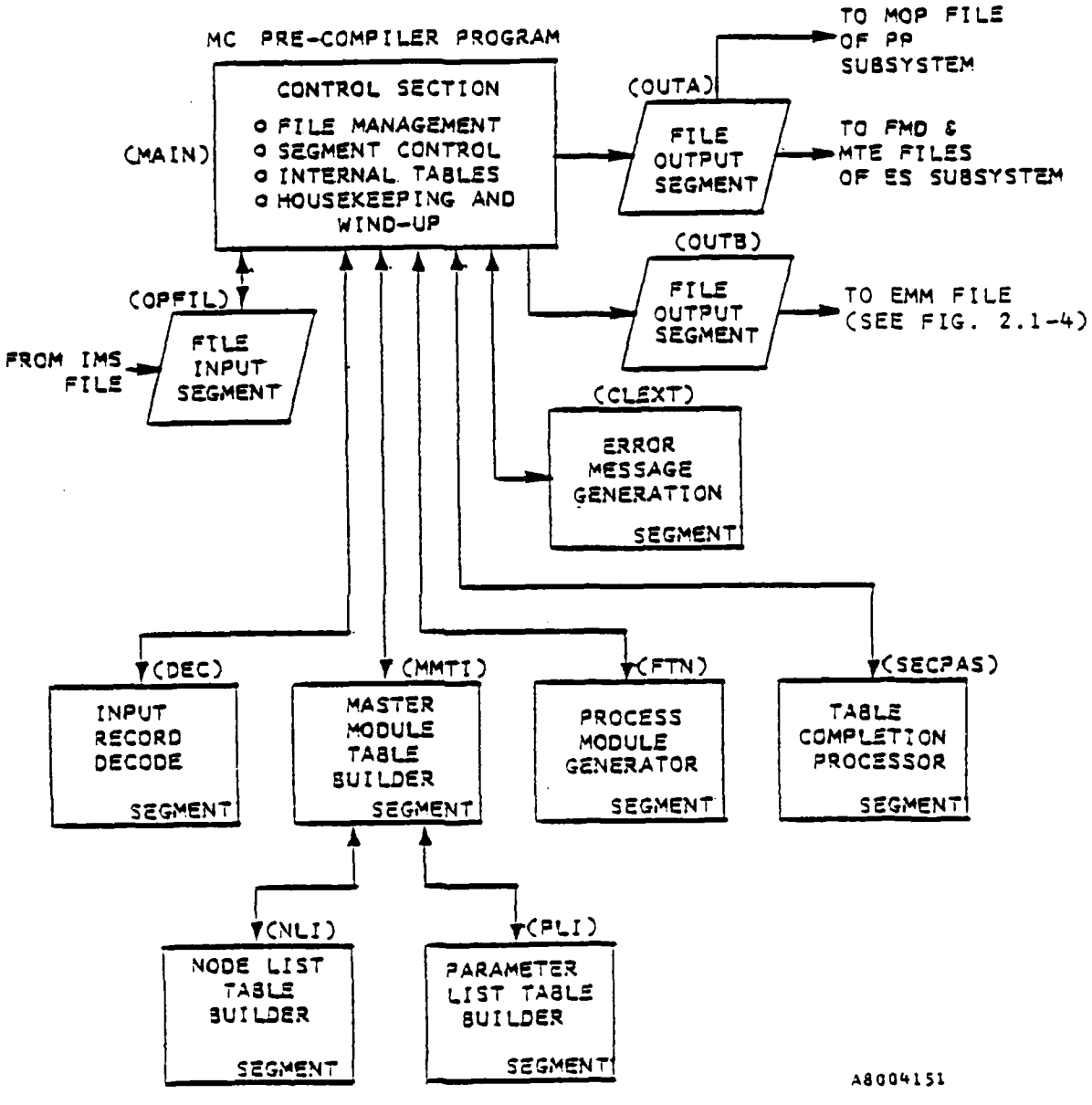


Figure 3-11. Macro-Level Flowchart of MC Pre-Compiler (MCP) Program

Pass B completes the table-building process, and constructs the FMD file.

Entries for the EMM file are generated in both Pass A and Pass B.

The MCP employs program segments as shown in figure 3-11. The various segments indicated employ a FORTRAN subroutine structure and employ suggestive subroutine names as indicated. The charts of figures 3-12 and 3-13 indicate the functional flow among the various program segments.

The MCP operates in "batch" mode, generating no interactive or on-line messages. All error conditions detected are posted to the EMM file for display and correction prior to execution of the ES subsystem.

The organizational hierarchy of the MCP program is depicted in figure 3-14.

3.2.3 Description of the General Target Simulation Model (TSM) Program

The ES subsystem provides the facilities to compose a TSM and to control TSM execution. A typical TSM consists of a fixed (ie, programatically constant) mainline (viz, the EK) and an automatically written, customized simulator module (viz, the PM) linked to it to form the complete customized TSM.

The elements which comprise the ES subsystem are shown in figure 3-5. The FMD file produced by the MC subsystem is submitted to the ICSSM host computer FORTRAN Compiler. By this mechanism, the model-specific PROCESS module is created in linkable, executable form, for incorporation into the TSM. In addition, the EK (FORTRAN-coded but submitted previously to the FORTRAN Compiler and available in a linkable, executable form) is invariant in every TSM.

The TSM uses the MTE file as input and produces three files as output:

- o The Execution Journal (EXJ) file.
- o The Signal Output (SIG) file.
- o The Event Journal (EVJ) file.

The TSM also employs the temporary I/O files - Signal Work (SW) and Coefficient Word (CW).

Program termination in the TSM program is automatic. When TSM program termination occurs, the EVJ file contents may be examined to determine if errors were detected during TSM program execution.

3.2.3.1 Major Operations Performed in the TSM Program

Organization and major functions in the TSM Program are depicted in figure 3-15. The individual macro-functions depicted are delineated in separate macro-level charts as:

(1) Control Table Management Portion - figure 3-16.

(2) Event Queue Table Management Portion - figure 3-17.

(3) Process Module Portion - figure 3-18.

The acronyms in parentheses in these figures are FORTRAN subroutine names used within the TSM.

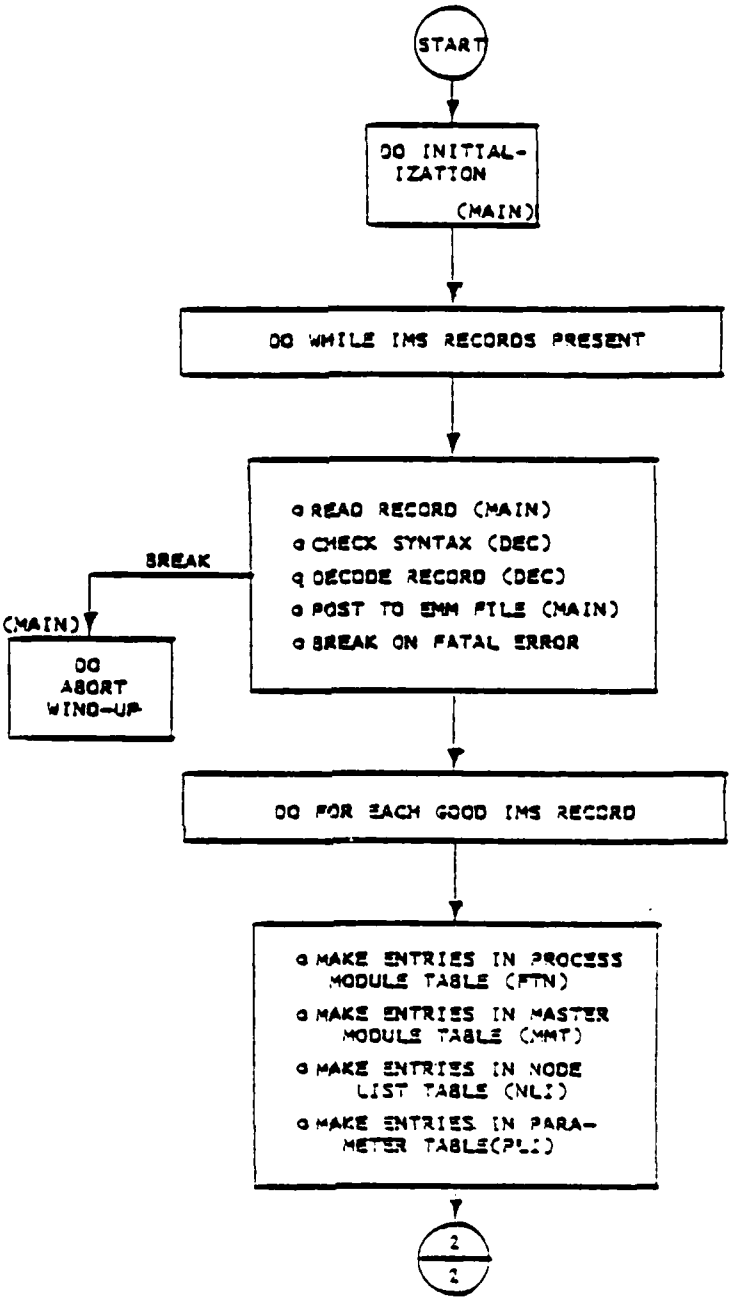


Figure 3-12. MC Pre-Compiler (MCP) Program Functional Flow (Pass A)

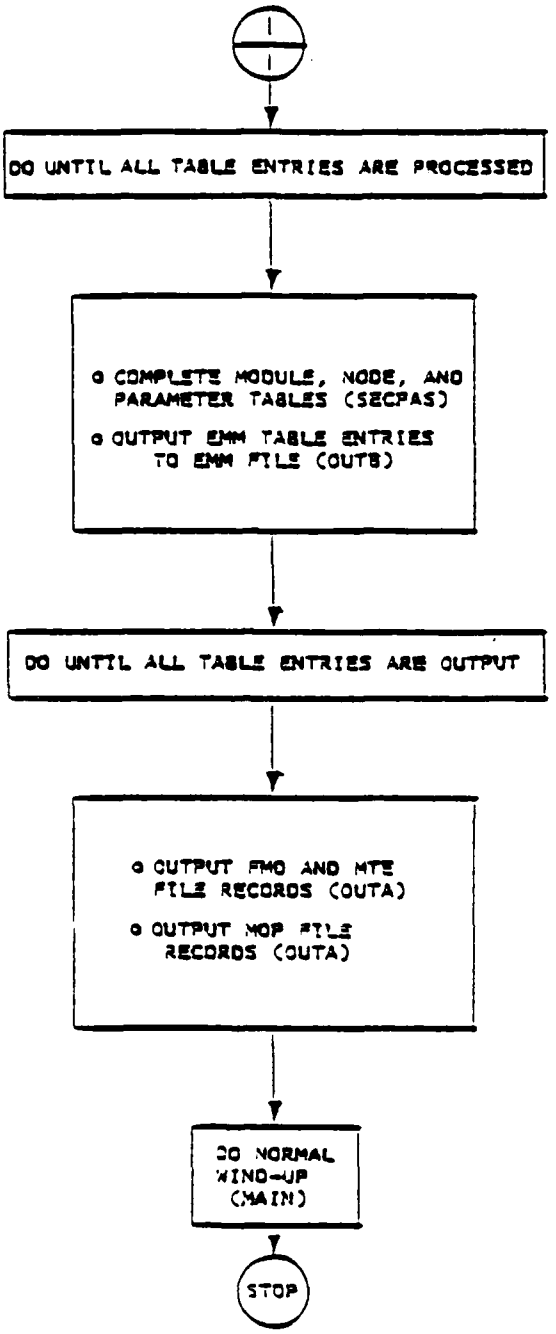


Figure 3-13. MC Pre-Compiler (MCP) Program Functional Flow (Pass B)

1. MAIN-----MAIN ROUTINE FOR PRE-COMPILER

    1.1 OPFIL-----OPEN FILES

    1.2 DEC-----READ IN OUTPUT PORTS AND DECODE

        1.2.1 CLEXT-----CLOSE FILES IF ERROR OCCURS

    1.3 MMT1-----CONSTRUCTION OF MASTER MODULE TABLE

        1.3.1 NL1-----PARTIALLY FILLS NODE LIST

        1.3.2 PL1-----FILLS UP PARAMETER LIST

        1.3.3 CLEXT-----CLOSE FILES IF ERROR OCCURS

    1.4 FTN-----CONSTRUCTS FORTRAN PROGRAM PROCES

    1.5 SECPAS-----FINISH CONSTRUCTING NODE LIST AND CONSTRUCTS TO-LIST

        1.5.1 CLEXT-----CLOSE FILES IF ERROR OCCURS

    1.6 CLEXT-----CLOSE FILES IF ERROR OCCURS

    (FILES ARE CLOSED NORMALLY AT THE END OF MAIN ROUTINE)

MCP Program Hierarchy

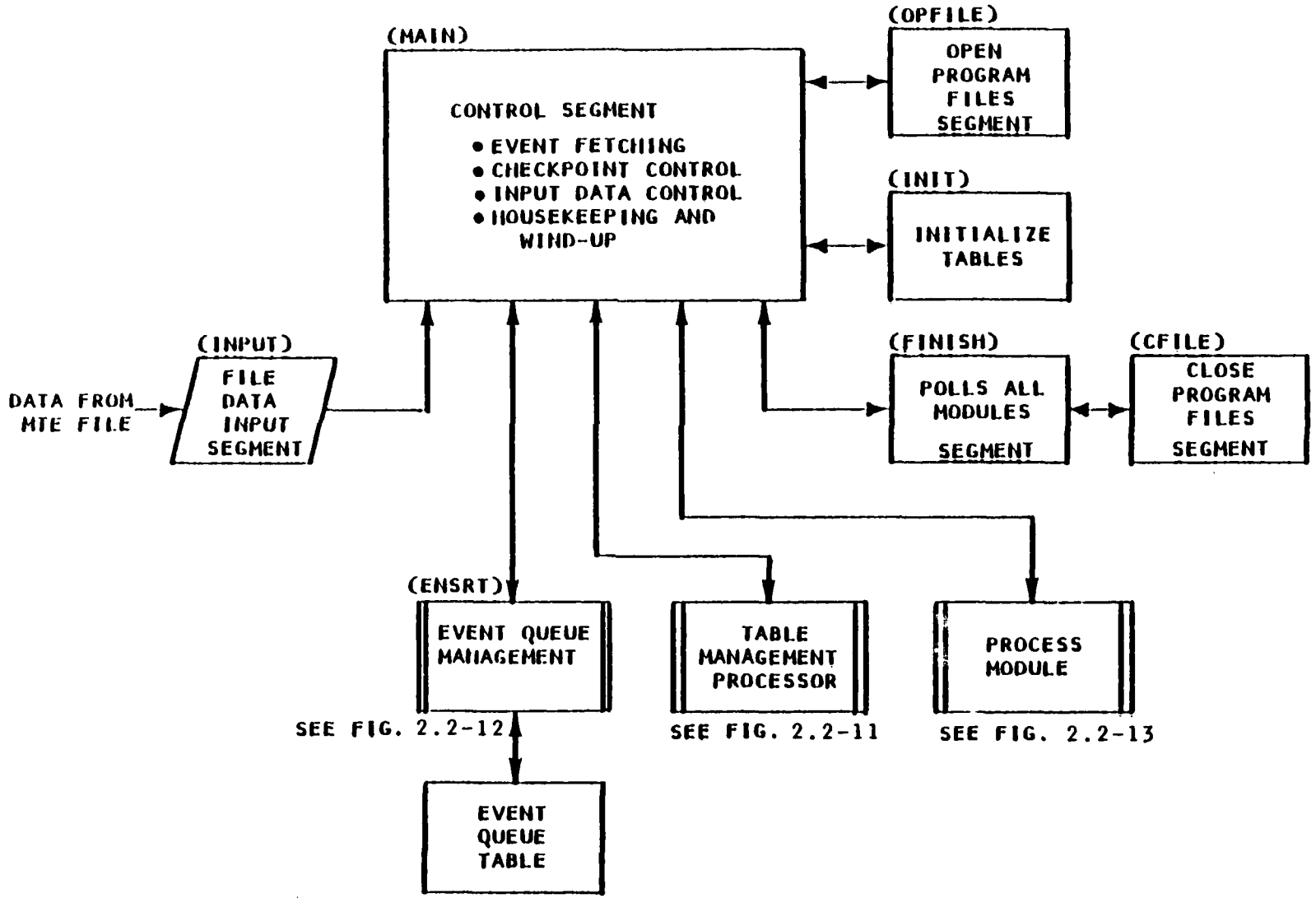


Figure 3-15. Macro-Level Flow Chart of TSM Program, Control/Executive (CE) Portion

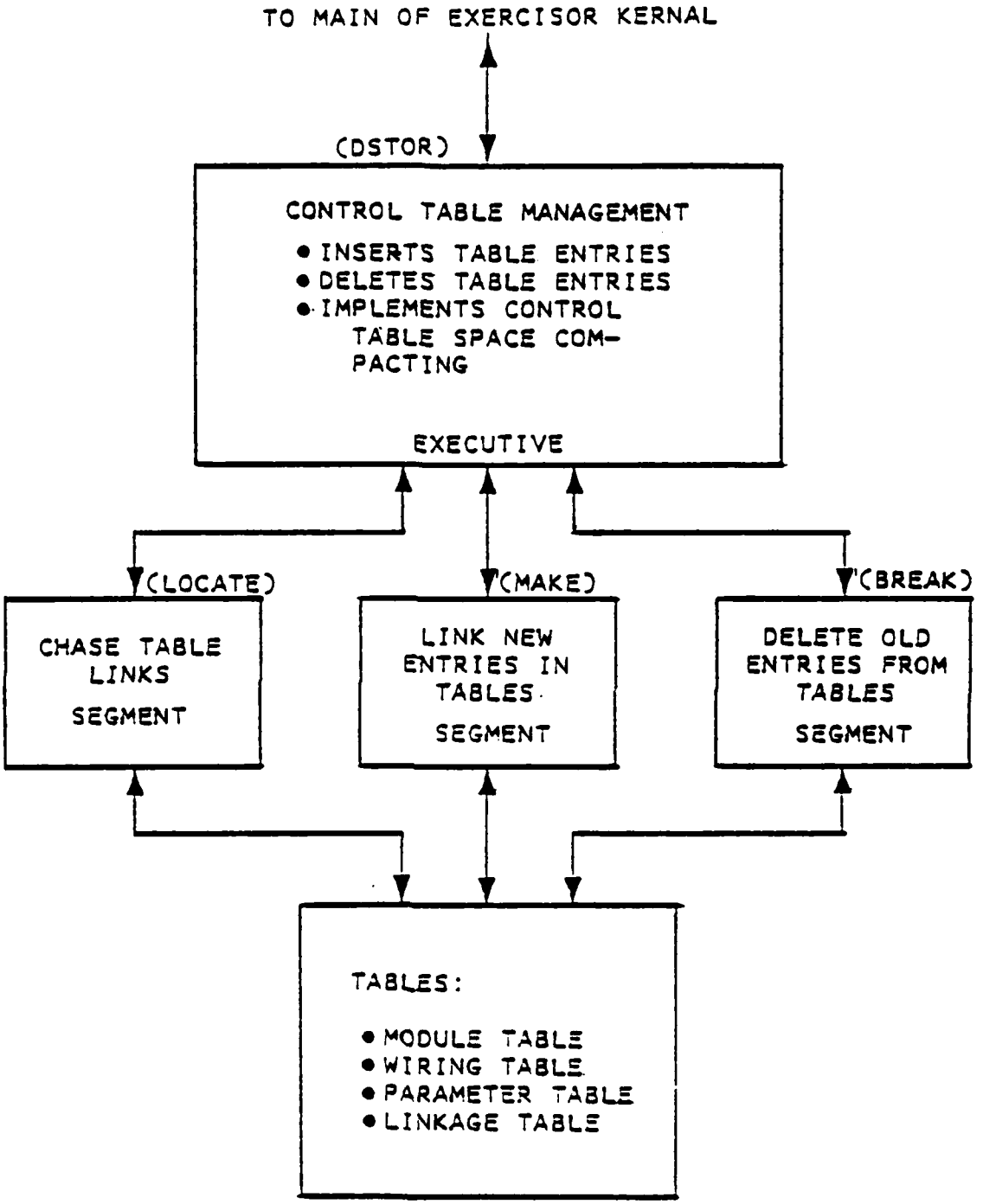


Figure 3-16. Macro-Level Flow Chart of TSM Program, Control Table Management (CTM) Portion

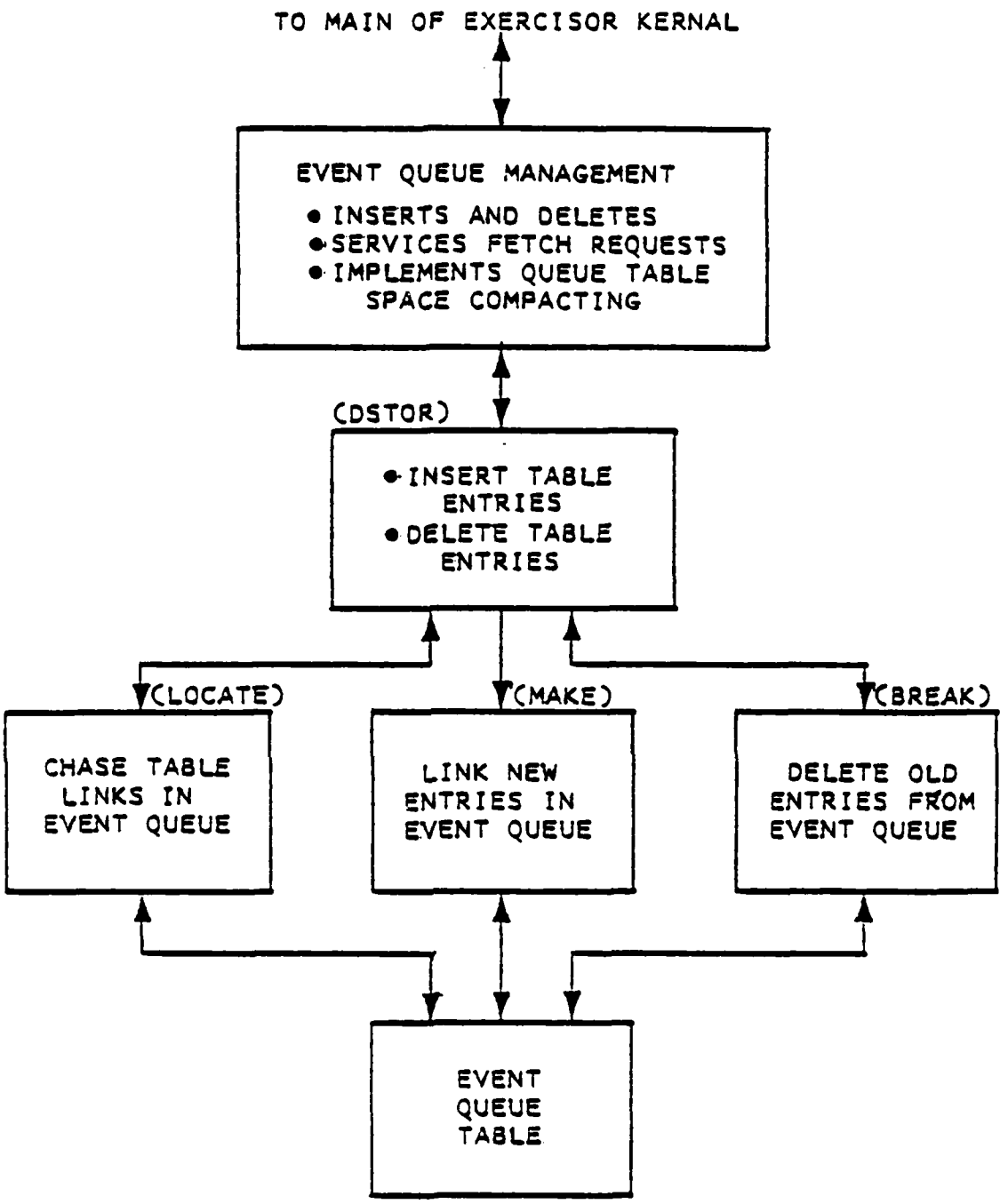


Figure 3-17. Macro-Level Flow Chart of TSM Program, Event Queue Table Processor (EQP) Portion

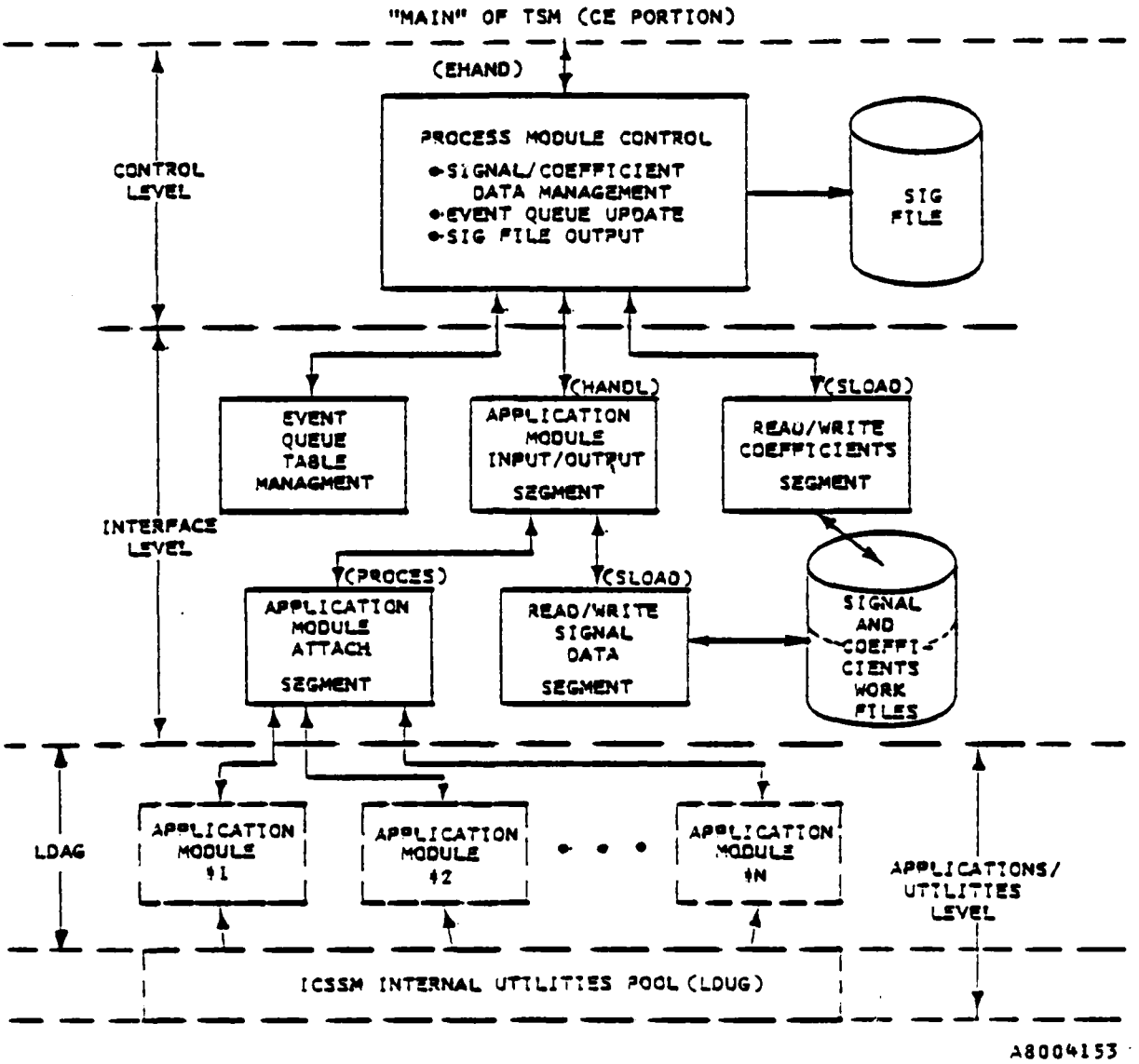
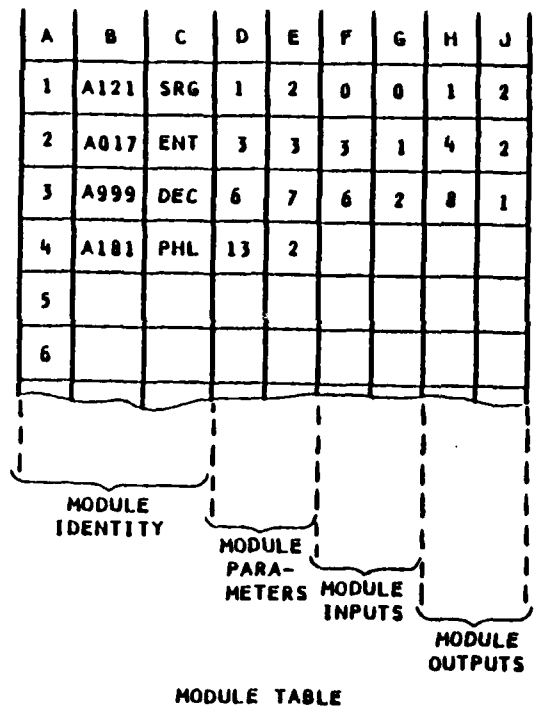


Figure 3-18. Macro-Level Flow Chart of Process Module

ADDRESS (IMPLICIT)	BACKWARD THREAD POINTER	FORWARD THREAD POINTER	QA QUEUE ORDERING (TIME/EVENT/NODE)	QB QC QD MEMBERSHIP LINKING POINTER
1	0	2	T_1	4
2	1	3	T_2	9
3	2	101	T_3	50
4	0	5	E_{11}	7
5	4	6	E_{12}	51
6	5	102	E_{13}	52
7	0	8	N_{111}	CRP_1
8	7	103	N_{112}	CRP_2
9	0	10	E_{21}	11
10	9	103	E_{22}	53
11	0	104	N_{211}	CRP_{99}
12				

Figure 3-19. Event Queue Table Organization, With Illustrative Examples



A. MODULE NUMBER (IMPLIED BY POSITION IN TABLE)

B. LIBRARY-ASSIGNED MODULE NAME

C. USER-ASSIGNED MODULE NAME

D. POINTER TO HEAD OF PARAMETER SET IN PARAMETER TABLE FOR GIVEN MODULE

E. NUMBER OF MEMBERS OF PARAMETER SET FOR GIVEN MODULE

F. POINTER TO HEAD OF NODE-CODE SET(INPUT NODES) IN NODE TABLE FOR GIVEN MODULE

G. NUMBER OF MEMBERS IN NODE-CODE SET(INPUT NODES) FOR GIVEN MODULE

H. POINTER TO HEAD OF NODE-CODE SET (OUTPUT NODES) IN NODE TABLE FOR GIVEN MODULE

J. NUMBER OF MEMBERS IN NODE-CODE SET (OUTPUT NODES) FOR GIVEN MODULE

Figure 3-20. Module Table Organization, with Illustrative Examples

K	L	M	N	P	Q
1	1	1	1	1	2
2	1	2	1	2	4
3	2	1	1	0	1
4	2	2	2	3	9
5	2	3	1	5	5
6	3	1	-1	0	2
7	3	2	-1	0	5
8	3	3	1	7	12
9	4	1	-1	0	8

NODE TABLE

K. NODE CODE (IMPLIED BY POSITION IN TABLE)

L. MODULE NUMBER

M. PORT NUMBER

N. NUMBER OF CONNECTIONS DEFINED FOR MODULE PORT (NEGATIVE VALUE SIGNIFIES PORT IS AN INPUT PORT)

P. POINTER TO BEGINNING OF MODULE-PORT-CONNECTED-TO THREAD IN TO-LIST TABLE

Q. MODULE PORT ASSOCIATION POINTER [IF MODULE PORT IS AN OUTPUT PORT, POINTS TO BEGINNING OF MODULE PORT SIGNAL-LIST-LOCATIONS THREAD IN SIGNAL/COEFFICIENT LOCATOR TABLE. IF MODULE PORT IS AN INPUT PORT, CONTAINS NODE CODE OF PORT FROM WHICH THE PORT CONNECTION AROSE]

Table 3-21. Node Table Organization with Illustrative Examples

S	R
1	3
2	6
3	17
4	20
5	7
6	10
7	9
8	
9	
10	

R. CONTAINS NODE CODE WHICH MAY BE USED AS POINTER TO FIND POSITION (K ENTRY) IN NODE TABLE AT WHICH CONNECTED-TO-MODULE/PORT INFORMATION IS LOCATED

S. (IMPLIED) ADDRESS OF MODULE-PORT-CONNECTED-TO INFORMATION. THIS ADDRESS IS POINTED-TO FROM COLUMN P OF NODE TABLE.

NOTE: THE VALUE CONTAINED IN COLUMN N OF NODE TABLE, IF GREATER THAN 1, DEFINES THE NUMBER OF SEQUENTIAL TO-LIST TABLE ENTRIES WHICH ARE NEEDED TO DESCRIBE FAN-OUT CONNECTIONS SO SIGNIFIED.

Figure 3-22. To-List Table with Illustrative Examples

T	U
1	P_{11}
2	P_{12}
3	P_{21}
4	P_{22}
5	P_{23}
6	P_{31}
7	P_{32}
8	P_{33}
9	P_{34}
10	P_{35}
11	P_{36}
12	P_{37}
13	P_{41}
14	P_{42}

T. (IMPLIED) ADDRESS OF PARAMETER VALUE ASSOCIATED WITH MODULE IN MODULE TABLE. THIS ADDRESS IS POINTED-TO FROM COLUMN D OF MODULE TABLE

U. A PARAMETER VALUE

Figure 3-23. Parameters Table with Illustrative Examples

V	W	X	Y	Z
1	9	13	T_{32}	12
2	0	3	T_{11}	1
3	2	7	T_{12}	4
4	0	6	T_{21}	2
5	0	8	T_{21}	5
6	4	10	T_{42}	3
7	3	0	T_{13}	6
8	5	0	T_{22}	8
9	0	1	T_{31}	10
10	6	0	T_{51}	7

SIGNAL LOCATOR

α	β	γ
1	0	4
2	0	3
3	2	7
4	1	6
5	0	8
6	4	0
7	3	0
8	5	0
9	14	15
10	0	12

COEFFICIENT LOCATOR

V. (IMPLIED) ADDRESS OF HEAD OF SIGNAL-RECORD THREAD. ALSO IS RECORD NUMBER FOR SIGNAL WORK FILE. USED AS POINTER IN COLUMN Q OF NODE TABLE.

W. POINTER TO PREVIOUS BEAD ON SIGNAL-RECORD THREAD

X. POINTER TO NEXT BEAD ON SIGNAL-RECORD THREAD.

Y. SIMTIME VALUE ASSOCIATED WITH SIGNAL-RECORD AT TABLE ADDRESS. COLUMN Y VALUES ARE IN MONOTONIC NON-DRECEARSING ORDER FOR EACH THREAD

Z. ADDRESS OF COEFFICIENT LOCATOR TABLE ENTRY α FOR COEFFICIENT RECORD ASSOCIATED WITH SIGNAL-RECORD. ALSO IS RECORD NUMBER FOR COEFFICIENT WORK FILE

α SIMILAR TO V. BUT FOR COEFFICIENT RECORDS.

β,γ SIMILAR TO W, X, BUT FOR CO-EFFICIENT-RECORD THREAD

Figure 3-24. Signal/Coefficient Location Table Organization with Illustrative Examples

3.2.3.1.1 Operations Performed in the TSM, Control/Executive (CE) Portion

The TSM program, CE Portion:

a. reads the contents of the MTE file, and transfers the values provided by the MTE file data to in-main-storage tables corresponding to those of figure 3-19 through 3-24 .

b. initializes data for the Event Queue table and the other in-main-storage tables.

c. processes Event Queue entries according to given rules, and maintains the Event Queue in proper order as relevant occurrences within the simulation are encountered.

d. controls the transfer of Signal and Coefficient data blocks to and from SW and CW files and to the SIG file.

e. oversees the transfer of data and control elements to and from the Process Module Portion.

3.2.3.1.2 Operations Performed in the TSM, Control Table Management (CTM) Processor Portion

The TSM program, CTM portion operates as indicated in figure 3-16. The CTM portion:

a. provides a means of controlling the interface between the CE portion and TSM control tables.

b. provides a means to locate and retrieve records in SW and CW files which are pertinent to the Event Code type and associated Node code for the "energized" applications module invoked by the Event Queue/Event Queue Processor action.

c. inserts new entry(s) in the relevant control tables based upon internal action of the "energized" applications module.

d. deletes entry(s) from the relevant control table based upon internal action of the "energized" applications module.

3.2.3.1.3 Operations Performed in the TSM, Event Queue Table Processor (EQP) Portion

The EQP portion:

a. inserts and deletes events from the Event Queue table.

b. maintains a minimum inventory of empty or non-utilized Event Queue table locations.

c. interfaces to the TSM-CW event-fetch segment, and services the fetch request.

3.2.3.1.4 Operation Performed in the TSM, Process Module (PM) Portion

The PM portion operates on three levels:

- o the Control level

- o the Interface level
- o the Applications/utilities level

The Control Level provides:

a. interface to the TSM, CE portion.

b. output of signal and coefficient records to the SIG file from the CW and SW files.

c. interface to the EQP portion whenever an "energized" applications module issues an "update" request.

d. communication with the Interface-level segments of the PM.

The Interface Level provides:

a. control of Signal List and Signal/Coefficient data transfer between Applications/utilities-level segments and TSM-CW COMMON areas.

b. transfer of state-parameter values and model-specified parameter values to and from the "energized" applications module via the Applications/utility-level segments.

The Applications/Utilities Level provides:

a. subroutine link and data/parameter processing which is manifested in the applications library modules "attached" via the customized PROCESS subroutine.

b. utility subroutine support from subroutines used in common among applications module algorithms to effect commonly required control and data manipulations.

The relationships which prevail for Control-level/Interface-level coupling are depicted in figure 3-25.

The functional relations which prevail for Interface-level/Applications-level coupling are depicted in figure 3-26.

The TSM program internal utility functions (available from the ICSSM LDUG) are depicted in figure 3-27.

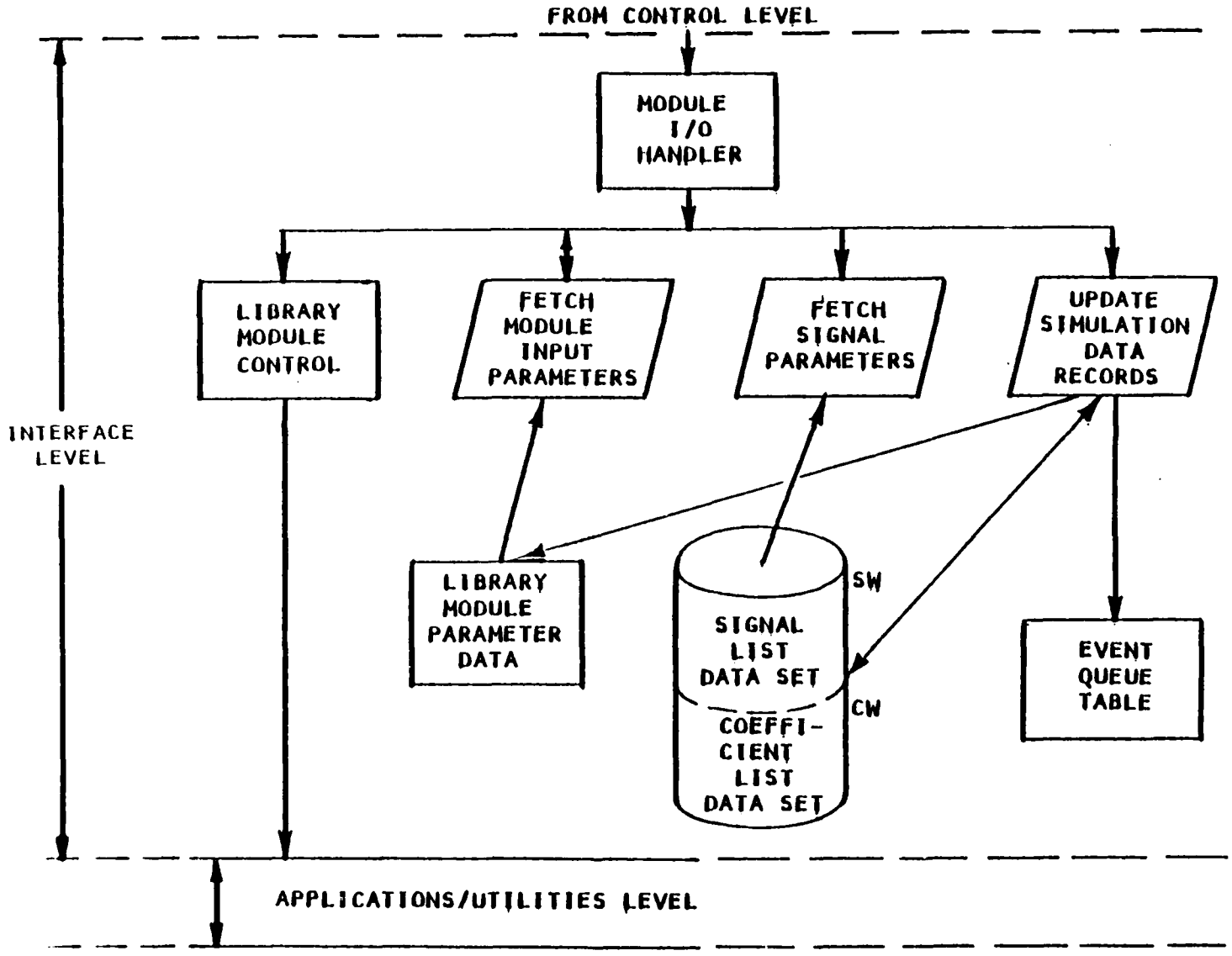
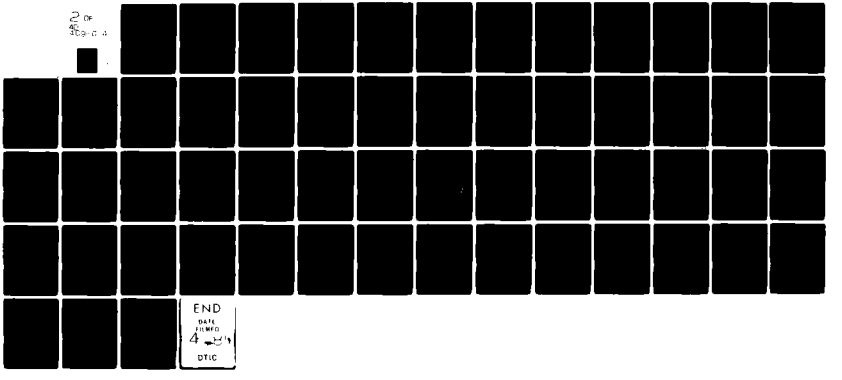


Figure 3-25. Control-Level/Interface-Level Processing, in Process Module

AD-A096 054 HAZELTINE CORP GREENLAWN NY F/G 17/2
INTERACTIVE COMMUNICATIONS SYSTEMS SIMULATION MODEL (ICSSM),(U)
DEC 80 J FAOLICELLI, W F GRIESE F30602-78-C-0197
UNCLASSIFIED 6410 NL



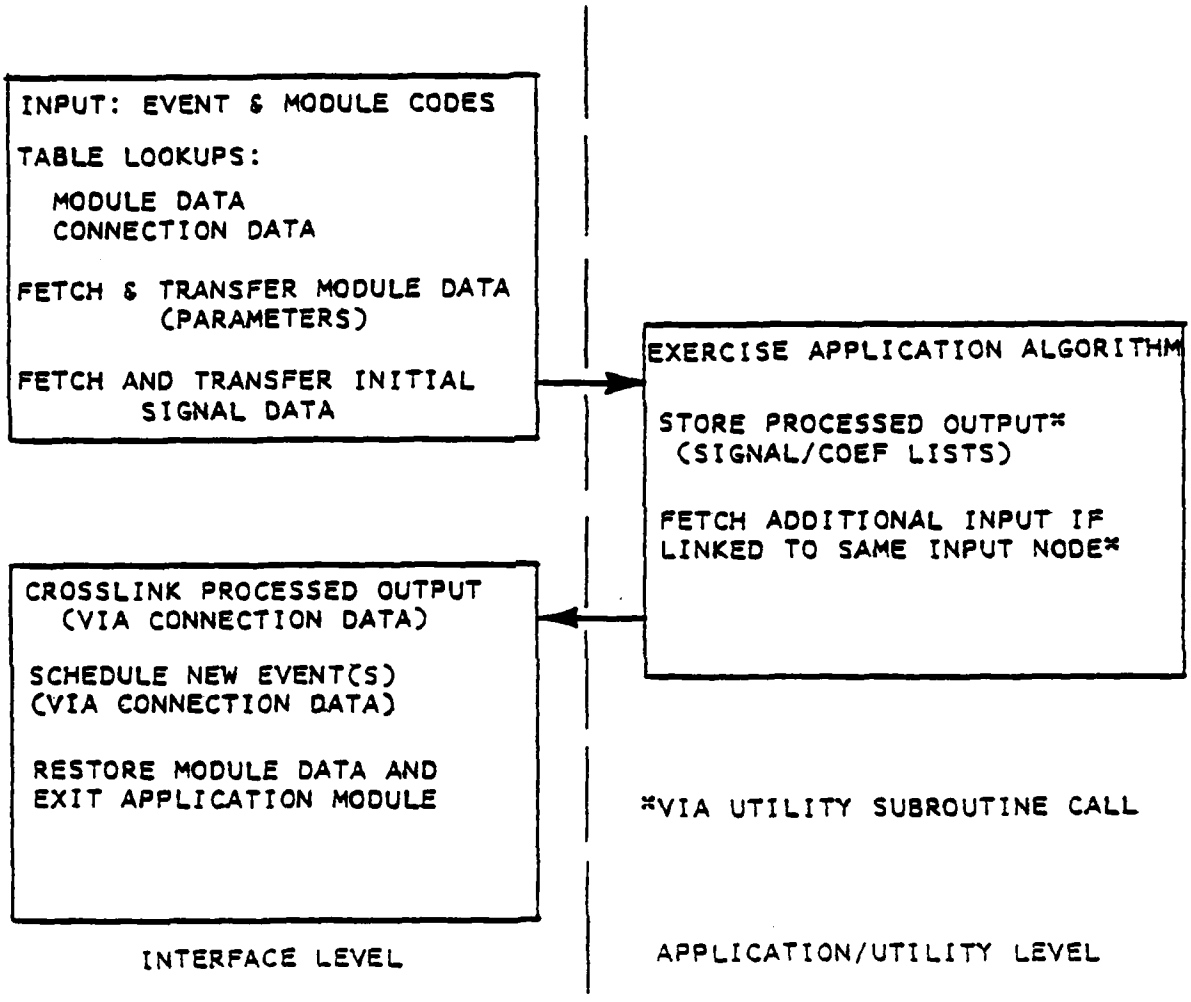


Figure 3-26. Interface-Level/Applications-Level Coupling, in Process Module

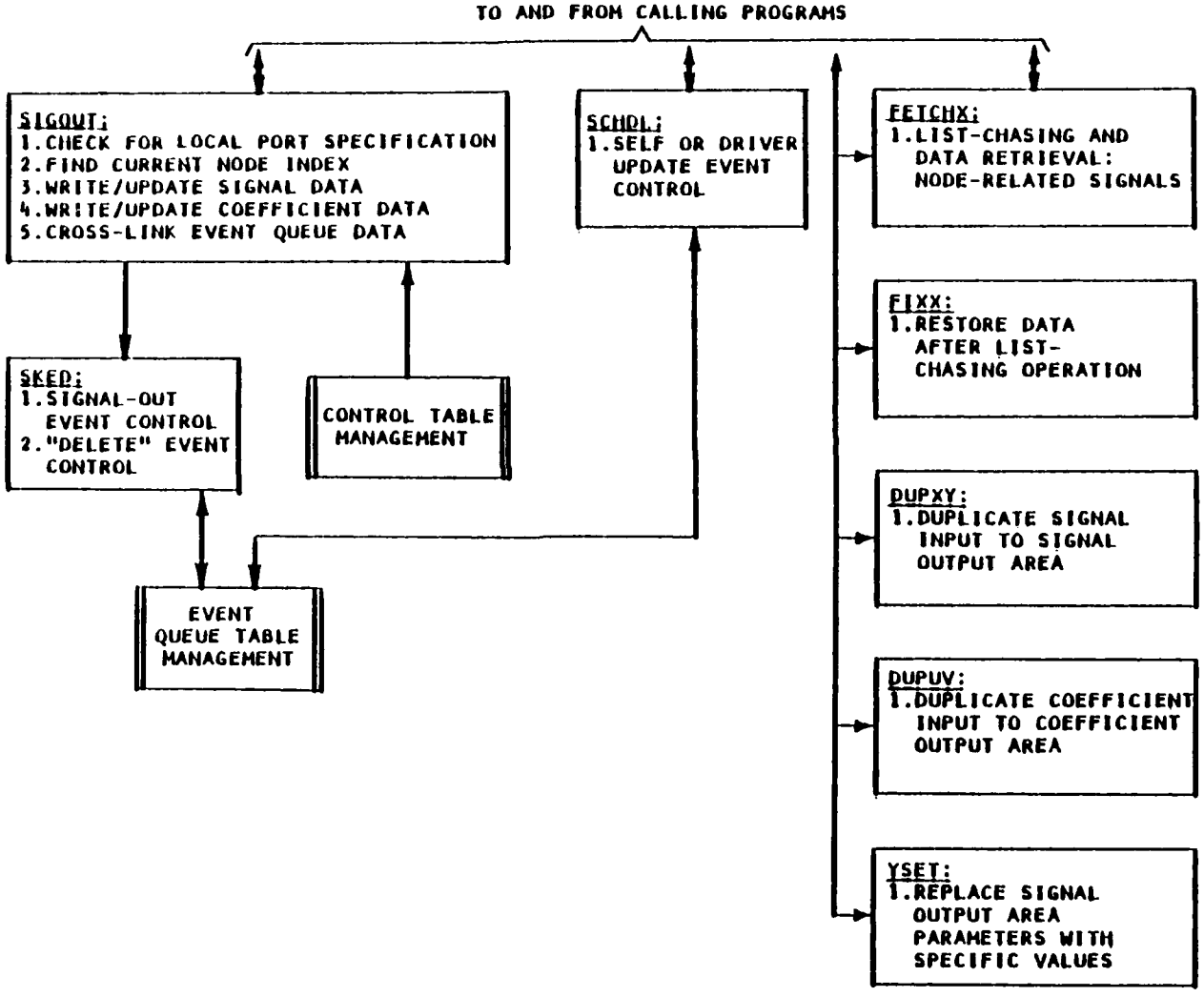


Figure 3-27. Required Internal Utilities

3.2.3.2 Internal Tables Employed in the TSM Program

TSM employs six internal tables:

- o Event Queue table
- o Module table
- o Node table
- o To-list table
- o Parameters table
- o Signal/Coefficient Location table.

The Event Queue table is processed by the Event Queue Processor. The remaining tables are processed by the Control Table Management Processor.

The six internal tables are inter-related as in the diagram of figure 3-28. This figure depicts the inter-table linkage scheme established within the TSM program and also depicts the relationship between the Event Queue table and the Control tables.

3.2.3.2.1 Event Queue Table

The form of the Event Queue table is shown in figure 3-19. This table is addressable by "slot" number. The table address number is implicit in the ordering of the table. Each address contains four "components" (columns QA, QB, QC, and QD of the figure). The QC column contains numbers representing any of three kinds of quantity, namely, values of SIMTIME, EVENT CODE, or NODE CODE. The Queue is "threaded" by pointers contained in columns QA and QB. Each "thread" strings like quantities together, as defined by the QC column. For example, a "thread" contains only values of SIMTIME, or only values of NODE CODE. More than one of each kind of thread may exist. Each such thread starts with a value of zero stored in the QA column. Succeeding "beads" on the thread are pointed-to-by the component value in column QB of

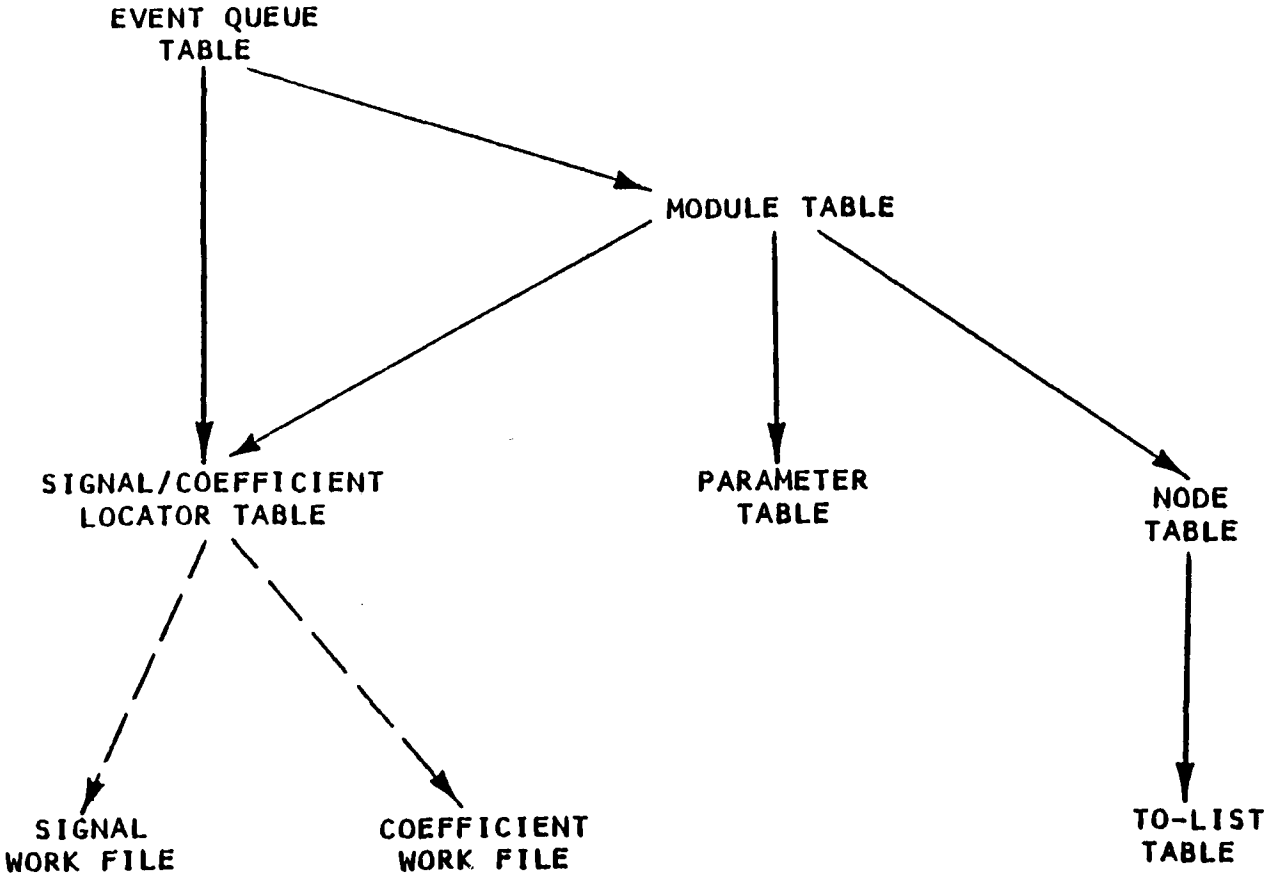


Figure 3-28. Internal Table Inter-Relationships

the same address. For example, in figure 3-19, addresses 1, 2, 4, 9, and 11 all contain a zero in column QA, signifying the head of, or starting point for, five different threads. The thread starting at address 1 contains only values of SIMTIME, ie, it threads through addresses 1, 2, 3 and 101. Similarly, addresses 4, 5, 6 and 102 thread through values of EVENT CODE only.

Threads formed in the manner described are also linked together by the values stored in column QD. QD contains values pointing to addresses which establish three links. The links so formed associate a value of SIMTIME with EVENT or EVENTS and with NODE or NODES. For example, in figure 3-19 the QD column component in address 1 contains the value 4. This points to address 4. The QD component in address 4 contains the value 7, which points to address 7.

Link elements form a "chain" comprising a value of SIMTIME, an EVENT CODE value, and a NODE CODE value. The last element (ie, the link terminator) resides in the QD column, and points to the Signal record, in the SW file which is associated with the SIMTIME, EVENT CODE, NODE CODE triplet on that link.

The Event Queue Management Processor establishes and maintains the internal contents of the Event Queue table.

The TSM Program, Control/Executive portion maintains a table pointer (acronym INITA) which points to the next event-time associated with the current event being processed from the Event Queue. The INITA pointer provides means to "chase" the "chain" of the internal control table in the order: Event Time → Event Code → Node Code → Node Code Pointer. The chase provides the data needed to hand control to the Process Module portion to effect the simulation processing dictated by the event retrieved and the node involved.

3.2.3.2.2 Simulation Control Tables

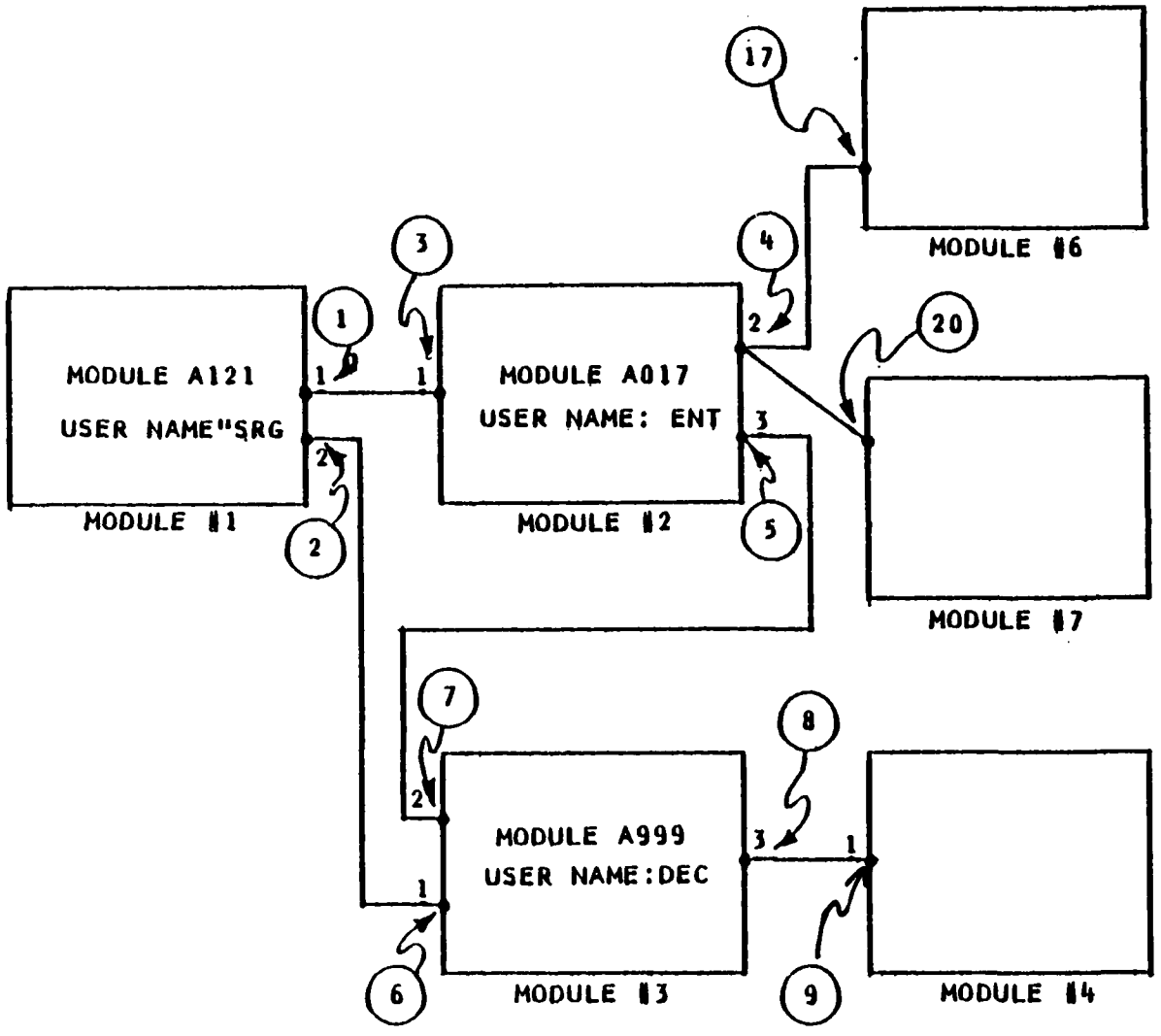
The Simulation Control tables (excluding the Event Queue table) control the internal operation of the TSM. These tables are implemented as in figures 3-20 through 3-24. These figures describe the nature of the various table entries, and the interrelationships and intra-relationships among them. These figures also contain specific entry values which depict the intended states of the tables with respect to a specific example shown in the block diagram of figure 3-29.

The Control Table Management Processor establishes and maintains the internal contents of the Simulation Control Tables.

3.2.3.2.3 TSM Program Event Codes and Signal List Elements

The Event Codes which appear in column QC of the Event Queue Table are defined in the table below:

Event Code Acronym	Event Code Value	Meaning
ION	1	An application module of the TSM must be turned on or has been turned on.
IUP	2	An application module output signal is available to update the SW file.
IOFF	3	An application module of the RSM must now be turned off or has been turned off.
ICON	4	An application module control signal has changed state and requests service.
IDEL	5	Delete a signal record from the SW file
IEND	6	The simulation termination event initiates a poll of ESK control tables for quiescence and to exit simulation.



NUMBERS WITHIN CIRCLES ARE NODE-CODE VALUES ASSIGNED IN ILLUSTRATIVE EXAMPLE

Figure 3-29. Model Used for Illustrative Examples, Block Diagram

Event Code Acronym	Event Code Value	Meaning
ICHK	7	Not implemented (reserved for later use in check-pointing).
ISTOP	8	Not implemented (spare).

The Event Codes are listed there in the order: highest priority event type first (ie, Event Code 1), lowest priority event type last (ie, Event Code 8).

The Signal List Record is a collection of descriptors applicable to application modules. Each execution of an application module is accompanied by an updated Signal List Record for that execution. However, cases of an empty or "null" Signal List can arise (ie, cases where all results of execution of a particular application module are reflected in the associated Coefficient Record only).

The fields (elements) of a Signal List Record will be as below:

FIELD ACRONYM	MEANING
TON, TOFF	SIMTIME values when application module was "energized" and was "de-energized". Such an interval is referred to a "segment".
FZ	Center Frequency (MHz)
BW	Signal Bandwidth value (MHz)
TDUR	Duration of module activation (ie, segment duration) (μs)
WVFM	Carrier Waveform name or type code
SIGV	A signal voltage (volts)
TZ	Time of Transmission (μs)
TERR	Transmitter Clock Error (μs)
ID	Transmitter ID Number

FIELD ACRONYM	MEANING
R	Transmitter-Receiver horizontal range (nmi)
ELEV	Elevation angle relative to Receiver (degrees)
AZIM	Azimuth angle relative to Receiver (degrees)
PROP	Propagation Factor
AMPL	A signal amplitude or signal power value (volts or watts)
PHAZ	A signal phase value (degrees)
DPLR	A doppler rate or value (knots)
NCOF	Number of samples in Coefficient record
SCAL	A scale factor for Coefficient record
SPAC	Code for Coefficient basis-space
DELT	An incrementing value of time between Coefficient samples (μs)
NODE	Node Code for associated output port
NMBR	(as required)
TOK	T_{OK} for module SIMTIME
ENCD	(as required)

3.2.4 Description of the Post-Processor Selector (PPS) Program

The PPS program provides means with which to specify the post-simulation signal-data reduction and the processing to be done on the data obtained from the execution of an ICSSM TSM. The PPS program upon initiation, prompts the user to provide the specifications needed for post-processing actions. After the prompt/response session is completed, the PPS program writes the required specification data to the Post-Processor List (PPL) file and terminates.

A macro-level flow chart reflecting the operations performed in the PPS program is provided in figure 3-30.

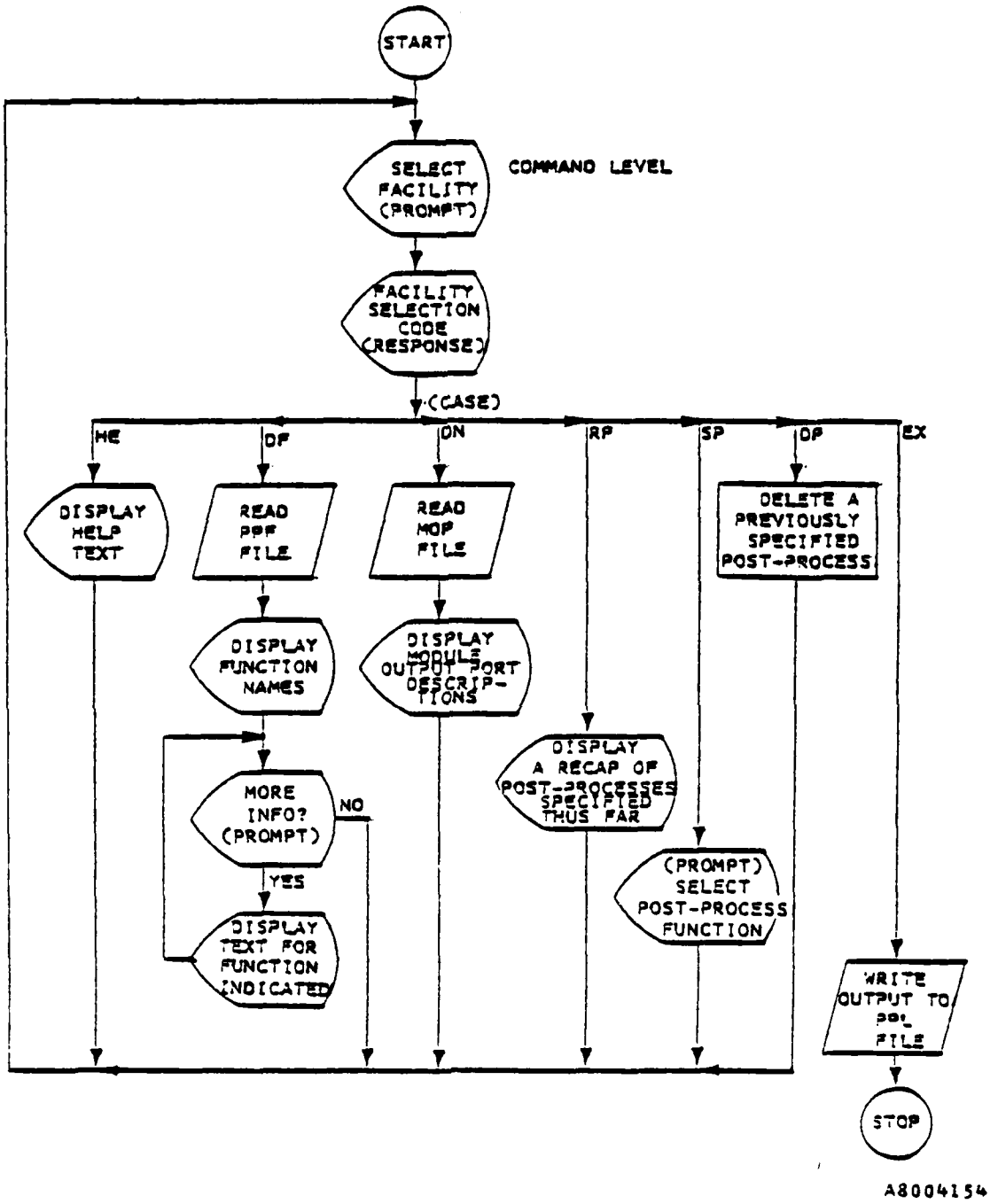


Figure 3-30. Macro-Level Flow Chart for Post-Processor Selector (PPS) Program

Hierarchic organization of the PPS program is depicted in figure 3-31.

The PPS program:

a. allows the user to select from among the post-processing functions available within the PPF file of the LD subsystem.

b. accepts user selection of the port-output-signals of the TSM to which the selected post-processing functions will be applied.

c. prepares a data file for input to the PPE program which directs the execution of the specified post-simulation processing operations.

3.2.5 Operation of the PPE Program

The PPE program operates according to the macro-level flow chart of figure 3-32.

PPE program operation terminates automatically, under normal conditions.

3.2.5.1 Major Operations Performed in the PPE Program

The organizational hierarchy for the PPE program is shown in figure 3-33.

1.0 initt -- initialize the plot-10 graphic package.

2.0 term -- inform plot-10 package that the terminal is a tektronics 4014.

3.0 chrsiz -- set the terminal character size (plot-10)

4.0 page -- clear and segment the screen into two parts.

- 4.1 newpag -- clear the screen (plot-10)
- 4.2 linhgt -- return the number of raster units per line (plot-10)
- 4.3 movabs -- move the graphic cursor (plot-10)
- 4.4 drawabs -- draw a line with the graphic cursor (plot-10)
- 4.5 anmode -- flush the graphic buffer and return crt to alphanumeric mode

5.0 nput -- handle a request for the output of a character string to the crt, clearing the screen when needed.

- 5.1 page -- (see 4.0)
- 5.2 output -- position the cursor and write a character string to the crt

6.0 he -- (help) output to the crt a text segment describing the use of this program.

- 6.1 page -- (see 4.0)
- 6.2 nput -- (see 5.0)

7.0 lf -- (list functions) output to the crt a listing of the available post processing functions.

- 7.1 page -- (see 4.0)
- 7.2 nput -- (see 5.0)
- 7.3 pftxt -- output to the crt descriptions of the post processing functions
  - 7.3.1 nput -- (see 5.0

Figure 3-31. PPS Program Hierarchy (Sheet 1 of 2)

8.0 ln -- (list nodes) output to the crt a listing of the available output ports. This list of read from a file produced by the pre-compiler.

8.1 page -- (see 4.0)

8.2 nput -- (see 5.0)

9.0 lp -- (list processes) output to the crt the specifications for processing to be done by the post processor.

9.1 page -- (see 4.0)

9.2 nput -- (see 5.0)

9.3 nfind -- locate a particular section of text in a text array.

9.4 concat -- copy one character string onto the end of another.

10.0 sp -- (specify a process) interactively specify a post processing function.

10.1 page -- (see 4.0)

10.2 nput -- (see 5.0)

10.3 gap -- interactively specify the ports at which the function is to be applied.

10.3.1 nput -- (see 5.0)

10.4 gparam -- interactively specify the function parameters.

10.4.1 nfind -- (see 9.3)

10.4.2 concat --(see 9.4)

10.4.3 nput --(see 5.0)

11.0 dp -- (delete a process) delete one of the specified post processes to be performed

11.1 nput -- (see 5.0)

12.0 ex -- (exit) write the specified processes into a file to be read by PPE program.

13.0 finitt-- terminate the plot-10 processing.

Figure 3-31. PPS Program Hierarchy (Sheet 2 of 2)

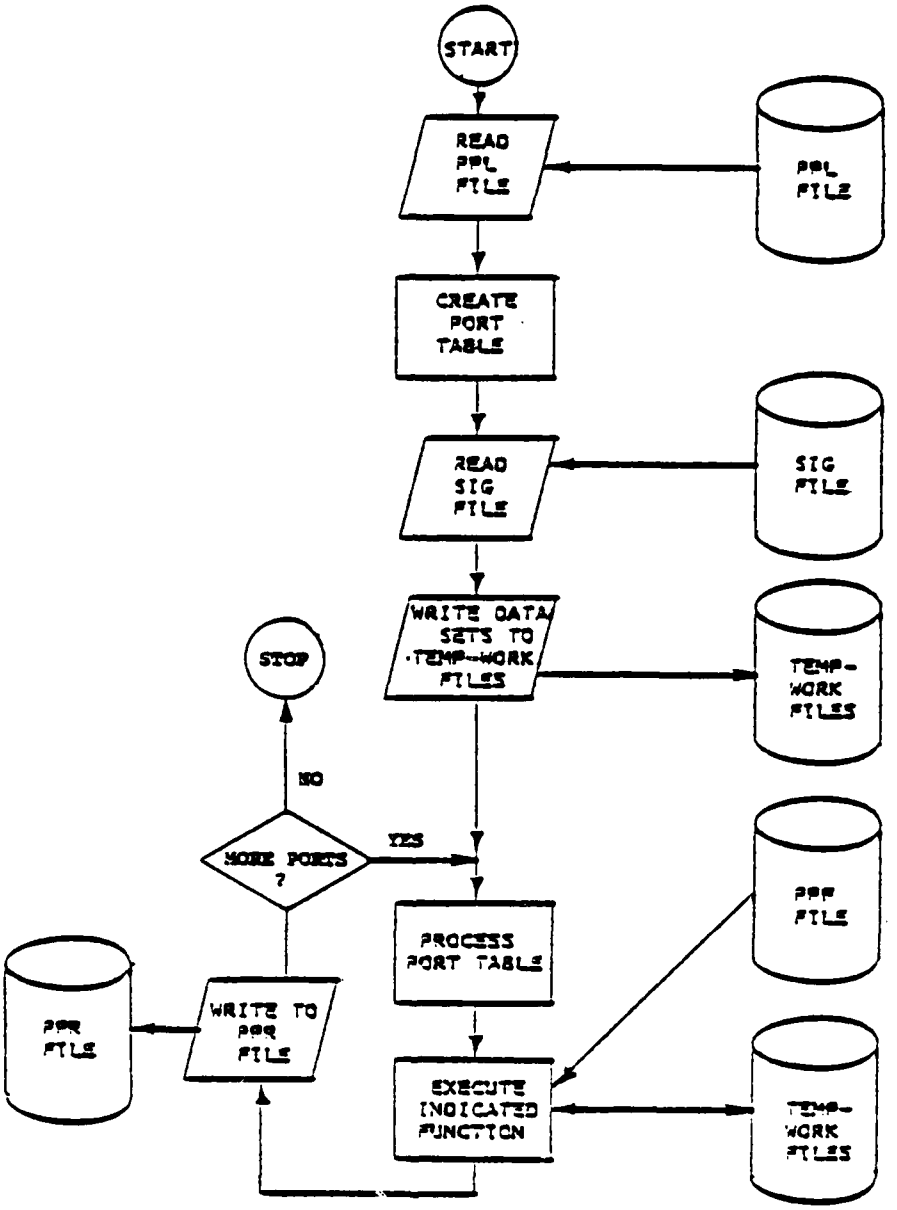


Figure 3-32. Macro-Level Flow Chart for PP Exercisor (PPE) Program

1.0 mklist -- go through the specified processes and construct a list of all the application ports.

1.1 psort -- sort the application ports by module number and then by port number. Duplicates are eliminated.

2.0 mkfls -- go through the file exer4_dat and copy the required records into the temporary files. One file for each application port.

3.0 exec -- call the appropriate function with its corresponding temporary files and parameters.

3.1 dofft -- read the temporary file and set up a complex array to be passed to cfft2.

3.1.1 cfft2 -- do a forward or reverse Fast Fourier Transform on a complex array.

3.1.2 gout -- output the result of the FFT in tabular form.

3.2 dober -- read the temporary files and determine the bit error rate and inter-error statistics.

3.2.1 berinit -- initialize the variables used by dober.

3.2.2 mino -- return the smaller of two integer arguments.

3.2.3 inc -- increment a 2-integer counter.

3.2.4 conv -- convert a 2-integer counter into a real.

Figure 3-33. PPE Program Hierarchic Structure

3.3 DETAILS OF ICSSM APPLICATIONS LIBRARY COMPONENT (ALC) IMPLEMENTATION

The following paragraphs provide descriptions of the programs, subroutines and files comprising the ICSSM Applications Library Component. The programs described are:

- o Elements comprising the LDAG and the LDUG
- o Programs for Application Library Maintenance

3.3.1 Implementation of the ICSSM ALC Directory and Library Elements

The ICSSM Library/Directory contains the files depicted in figure 3-7.

The Library/Directory comprises eight data sets organized into an Applications Group (LDAG) and a Utilities Group (LDUG). The LDAG provides the structure for on-line access to the Applications Modules contained in the ICSSM Applications Library.

The functional elements (Modules & DMS) required for communications system modeling reside in the Library Module (LM) file. This is an ASCII file containing all subroutines which may be employed by the user in configuring the TSM.

The LM file contents are divided into "Chapters". The number of Chapters depends upon the taxonomic structure employed in classifying the Modules. The taxonomy is reflected in the contents of the Library Chapter (LC) file and the Library Chapter Detail (LD) file, in that the LC file contains data on the most general classification of library modules, and the LD file contains data describing those library modules comprising each of the general classes (or Chapters) delineated in the LC file. The structure of these two files is analogous to chapter headings and within-chapter details provided by the "Table of Contents" of most textbooks.

3.3.1.1 Directory Structure Implementation in LD Applications Group

The LD Applications Group (LDAG) is organized into three ASCII files which contain inter-related data (see figure 3-34).

a. File LC is a sequential file which contains one 80-position ASCII record for each Chapter in the LD sub-system. Each record contains an 18-character Chapter description

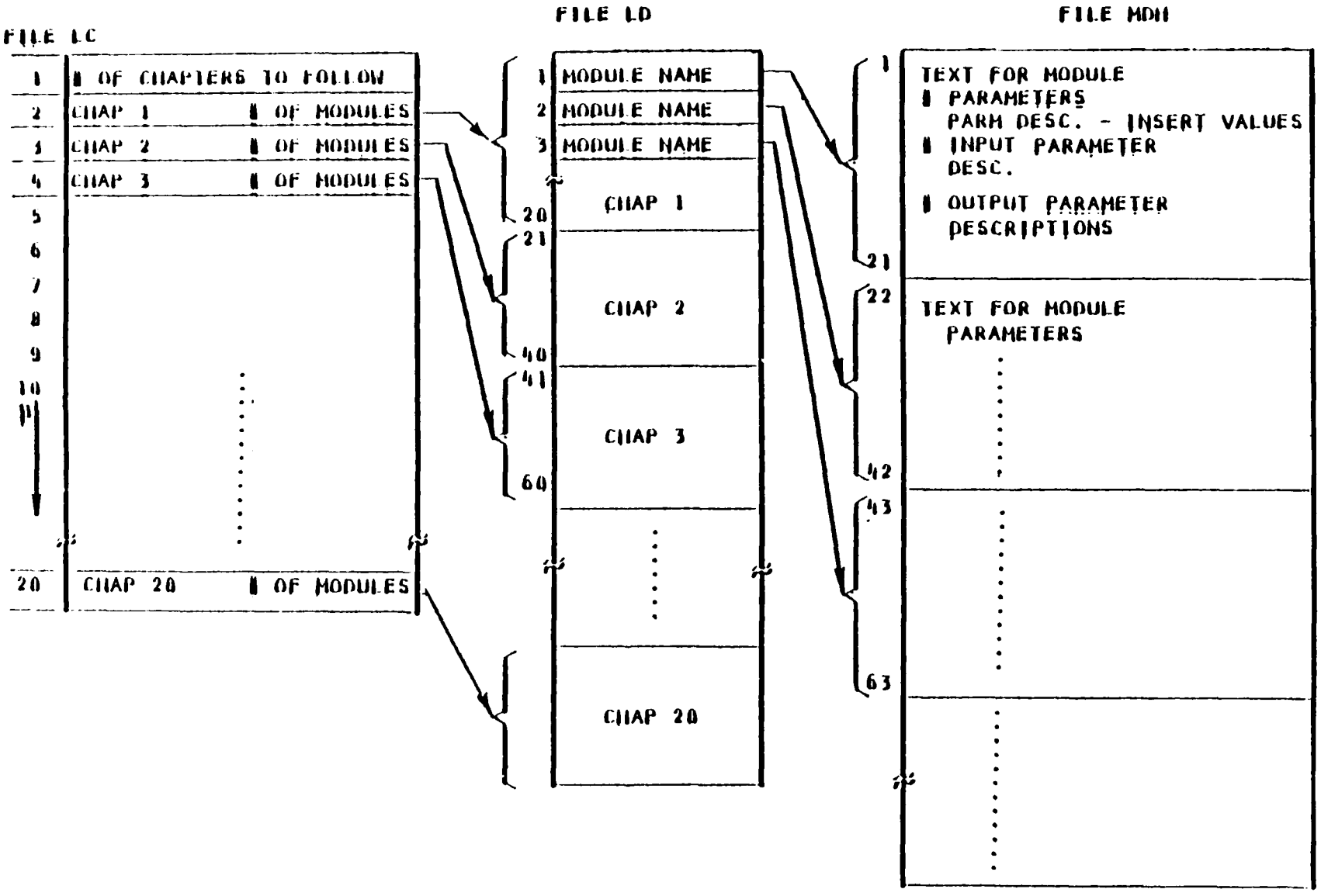


Figure 3-34. Inter-Relationship of Applications Library Directory Files

field and a 3-digit field containing the number of Application Library modules in that Chapter. The remainder of each record is blank.

b. File LD is a direct-access file containing 80 position ASCII records. Each record contains a 6-character module name field (the library-assigned name for the module). The remainder of the record is blank. There may be up to 20 such records, per Chapter record in the LC file.

c. File MDH is a random-acess file containing 80-position ASCII records. Each module record in the LD file generates a group of 21 records in the MDH file. Each such group is organized as in table 3-2.

Table 3-2. Module Description and Help File Data Items

Position # of Record within a Group	Description of Group Record Contents
1, 2, 3	Text describing nature of Module and method of use
4	Description of "hidden" parameters
5	Number of parameters required by Module, and number of subroutines required by Module.
6 thru 25	Descriptions and names of parameters for Module
26	Number of input ports defined for Module
27 thru 31	Descriptions of input ports of Module and instructions for their use
32	Number of output ports defined for Module
33 thru 37	Description of output ports of Module and instructions for their use
38	List of subroutine names CALLed by Module

3.3.1.2 Description of Library Module (LM) File Entries

The LM file is a user-defined file in the ICSSM host computer applications or user file area. It contains either FORTRAN or compiled (ie, relocatable) versions of each Applications Library entry, or both, depending upon user needs, upon ICSSM Librarian requirements, or upon host computer requirement. The LM file contains three types of entries:

- Those used as general-purpose modeling elements via the MC subsystem (ie, in the TSM).
- Those intended as general-purpose test modules for ICSSM maintenance purposes.
- Those intended for ICSSM validation tests (these may have general-purpose modeling application as well). All types of entries described above are referenced by the LD, LC, and MDH files.

3.3.1.3 Description of General Purpose LM File Items

The general-purpose Items (Modules) in the LM file are listed in tables 3-3 and 3-4.

Table 3-3. Applications Modules (Class I Items) In the LM File (Sheet 1 of 2)

Module Name	Module Function
SOURC2	A source of SFSK-modulated binary data.
CHNDIS	A model of a communication channel with additive band-limited gaussian noise.
SYNCMF	Model of a synchronized SFSK waveform-matched filter.
DECOD4	SFSK decoder/decision model.
BUFFER	A delay-line/storage-buffer model.

Table 3-3. Applications Modules (Class I Items) in the LM File (Sheet 2 of 2)

Module Name	Module Function
COMPA	A bit stream comparator/bit error computing module for use in test and validations.
SOURCE	A source of voice-emulation waveforms.
DETECT	An ideal envelope detector model.
COMP	A power spectrum difference comparator for use in tests and validations.
SOURC3	A source of random binary sequences (uniform PDF).
ENCOD3	A QPSK bit-stream-encoder.
MODUL3	A carrier-wave QPSK modulator.
CHNLC	An all-pass propagation channel model with additive correlated white gaussian noise.
PLL	A phase-locking loop model with correlated noise in the CW reference signal.
DEMOD	A demodulator of QPSK-modulated CW which includes the correlated noise due to synchronization errors in phase-locking loop.
DECOD3	A QPSK decoder model.
ADAPT	A general adaptive antenna model.
PROPL	A general propagation model which includes additive disturbances and multipath effects.
OTERM	A terminator for general use in modeling.

Table 3-4. Dependent Modeling Routines (Class 2 Items) in the LM File (Sheet 1 of 3)

Subroutine Name	Subroutine Function
XMTR (FZP, BWP, WVFMP, SIGVP, TERRP)	A routine for initializing channel parameters.
DPSKBT (SEED, OLDBIT NENBIT, PICBIT)	A binary DPSK modulation source.
SFSK (T,R,V, TREF, FMHZ, TERR,TOD,SI SQ)	A generator of sinusoidal frequency/ phase-shift keyed CW signal.
KODE (TX, TPD, AI, AQ)	A generator of in-phase and quadrature samples for the 'analytic signal'.
NOISE (V, N,) SIGMA, ISEED)	A source of white gaussian noise.
ERROR (MODN)	A routine for writing error messages to the EMM File.
INITB	An initializing routine used by module COMPA.
BFWRI	A routine used to transfer data within the TSM.
AM (AMOD, V,N)	A CW amplitude modulator.
CNOIS (SIGMA, ISEEDX, ISEEDY, TDECOR, XN, YN)	A source of time-correlated additive white gaussian noise voltage.
UNQPSK (KS, KC, A, B)	A QPSK demodulator routine.
INTGRT (M, NCOPF)	A general-purpose integrator routine.
CCVM (AR, AI, BR, IB, CR, CI, N)	Routine to multiply vector B by complex A.
WEIGHT (WI, WQ, NEL, G, TCONST, KOUNT, GT, ALPHA, TOT)	A routine to compute weight coefficients in adaptive antenna model.

Table 3-4. Dependent Modeling Routines (Class 2 Items) in the LM File (Sheet 2 of 3)

Subroutine Name	Subroutine Function
CUTVM (AR, IA, BR, BI, IB, CR, CI, N)	Routine to perform complex vector multiplication.
ALPAS	Routine for signal manipulation in all-pass propagation channel model.
GEOM (XTT, ITX, TNOW, R, PHI, S, THD, SDOT, SG, THG, SGDOT)	A routine to compute geometry of receiver signal effects based on transmitter/receiver antenna relationships.
POSIT (X, T)	Routine to update transmitter antenna position in doppler calculations.
DIRECT (R, TH, PG, SPRD, DT, VT, VR)	Routine to compute angles theta and phi of transmitter relative to receiver.
XANT (TH, PH)	Routine to compute flat-earth range.
RBLAD (TH, PH)	Routine to compute antenna element spacing in antenna array.
SPEC (R, S, TH, PH, SPRD, DT, VR, PSL, RHO, PSI)	Routine to compute effects of specular reflection in propagation channel model.
GREFL (SN, CN, F, AMAG, FAZE)	Routine to compute effects of ground reflection in propagation channel model.
ROUGH (SINTH, ALMBDA, RUFF, HT, HR, RH)	Routine to calculate roughness factor for ground reflection modeling.
FLUCT (RHOP, PHBP, EP, RHOEFP, PHBEFP)	Routine to model fluctuation of ground specular reflection multipath.
STORE (INDX, PR, SQ, THQ, PHI, SPRDQ, DTG, VRC, RHO, PSI, S, TH, PH, SPRD, DT, VT, VR, XT, TNOWP, THD, SDOT, SGDOT)	Modeling utility for storing module parameters into a working array.

Table 3-4. Dependent Modeling Routines (Class 2 Items) in the LM File (Sheet 3 of 3)

Subroutine Name	Subroutine Function
REFSFS (NRFBTS, REFAMP, IERC, REFU, SAVQTS, INSAMP, DELTCP, SAVSD, FMHZP, IPCPAT)	Generates SFSK-modulated reference modulation signal.
UNWIND (ARRAY, NCOFX, BARRAY, NELEM)	Routine to convert floated version of binary number to bit pattern of ones and zeroes.
CRRSMP (FMHB, CPHAS, NSAMP, TX, IERC, SEPOCH, CARRI, CARRQ)	Generates sample of in-phase and in-quadrature carrier.
CHIP (DELTC, KDWVFM, NSAMP, IERC, SMPCI, SMPCQ, SEPOCH)	Generates sampled SFSK chip-modulation-waveform.
MDULAT (TI, TQ, NSAMP, IERC, SQTS, CARRI, CARRQ, SMPCI, SMPCQ)	Generates SFSK-DPSK modulated carrier.
MDULAR (TI, TQ, NSAMP, IERC, SQTS, CARRI, CARRQ, SMPCI, SMPCQ)	Generates sampled bandpass correlation reference.
RESTOR (IDXP, RP, SQ, THQ, PHI, SPRDQ, DTG, VTG, BRG, RHO, PSI, S, TH, PH, SPRD, DT, VR, XT, TNOWP, THD, DCOT, SGDOT)	Modeling utility for storing working parameter values in parameter array.
RPIGRG (AMPI, N, M)	Normal random variable generator (zero-mean, unit-variance)

3.3.1.4 Description of SU File Data Items

The LD Utilities Group (LDUG) is organized into two files which contain FORTRAN subroutines used in supporting applications within the ICSSM system. The SU file contains the AP-120B Array Processor Emulator routines, and certain other routines used in several subsystems of the ICSSM system, and in the design of Applications Library modules. These are defined in tables 3-5 and 3-6.

Table 3-5. AP-120B Emulator Routines (Class 3 Items) in the SU File (Sheet 1 of 3)

Subroutine Name	Subroutine Function
VCLR (C,N,N)	Clears a vector array (CLEARS→C).
VMOV (A,I,C,K,N)	Moves a vector array (A→C).
VNEG (A,I,C,K,N)	Negates all vector components (-A→C).
VADD (A,I,B,J,C, K,N)	Adds two vector arrays (A+B→C).
VSUB (A,I,B,J,C, K,N)	Subtracts two vector arrays (A-B→C).
VSMULT (A,I,S,C, K,N)	Multiplies vector by a scalar (SxA→C).
VDOTPR (A,I,B,J, C,N)	Dot product of two vectors (A·B→C).
VLDG (S,I,C,K,N)	Base 10 log of a vector (LOG_{10}A→C).
VEXP (A,I,C,K,N)	Exponentiate a vector (EXP(A)→C).
VSIN (A,I,C,K,N)	Sine of vector (SIN(A)→C).
CVMUL (A,I,B,J,C, K,N, DUMMY)	Complex vector multiply (A/B→C).
VCOS (A,I,C,K,N)	Cosine of a vector (COS(A)→C).
VATAN (A,I,C,K,N)	Arctan of a vector (ARCTAN(A)→C).
VMUL (A,I,B,J, K,N)	Multiply two vectors (AxB→C).
VDIV (A,I,B,J, C,K,N)	Divide a vector by a vector (A/B→C).

Table 3-5. AP-120B Emulator Routines (Class 3 Items) in the SU File (Sheet 2 of 2)

Subroutine Name	Subroutine Function
VSQRT (A,I,C,K,N)	Square root of a vector ($\sqrt{A}\rightarrow C$).
SVE (A,I,C,N)	Adds vector components ($a_1+a_2+\ldots.\ a_1\rightarrow C$).
VFLT (J1, I, C, K,N)	Converts integer vector components to floating point representation (FLOAT (A)→C).
POLAR (A,I,C,K,N)	Rectangular to polar conversion of vector.
RECT (A,I,C,K,N)	Polar to rectangular conversion of vector.
CVMAGS (A,I,C,K,N)	Complex vector magnitude squared ($\|A\|^2\rightarrow C$).
MTRANS (A,I,C,K, NRC,NCC)	Transpose a matrix ($A^T\rightarrow C$).
MMUL (A,I,B,J,C, K,NRC,NCC,NCA)	Multiply two matrices ($A\times B\rightarrow C$).
CONV (A,I,B,J,C, K,N,M)	Convolve A with B (A*B→C).
CFFT2 (C,N,IF)	Fourier transform of array (F(C)→C').
VABS (A,I,B,J)	Absolute value of vector (ABS(A)→B).
MAXV (A,F)	Component of A having maximum value.
BITRV3 (C,N)	Rearranges coefficients obtained in AP-120B subroutine CFFT2.
APPUT (A,I,N, (IDUMMY)	Stores array A in array processor buffer.
APGET (A,I,N, IDUMMY)	Retrieves array A from array processor buffer.

Table 3-6. Simulator Internal Support Routines (Class 4 Subroutines) in the SU File (Sheet 1 of 2)

Subroutine Name	Subroutine Function
YSET (TIME, TDURP, DELTP, IDP, NMBRP, NCOFP, SPACP)	Sets up operating control values for on-coming TSM module.
SIGOUT (TIME,NOUT, NEW,KNTRL)	Transfers active TSM module output to EK COMMON area and creates output Event Queue entry.
INIT	Initializes pointers in Event Queue and control tables of TSM.
SCHDL (NODE,TIME)	Updates Event Queue table to schedule new event for subsequent TSM processing.
EHAND (ISIG)	Regulates passage of control between EK mainline and active TSM module.
HANDL	Loads and stores data between active TSM module and EK COMMON.
SLOAD (ISIG)	Transfer Signal List data from SW file record to EK main storage area.
DUPUV	Transfers data from UBLOCK to VBLOCK in EK COMMON area.
DUPXY	Transfers data from XBLOCK to YBLOCK in EK COMMON area.
FETCHX (NODE,INDX)	Chases time/event/node chain for Signal List pointer.
FIXX	Restores original contents of TSM XBLOCK COMMON after thread chase.
SKED (TIME,ISIG, LEVNT,JTO,KTO)	Updates/schedules events for self-driven or externally driven TSM modules.
CFILE	Closes all TSM files and terminates TSM operation.
DSTOR (M,R,INDX, INITL,ID,IREC, IFN,IDUP,IRTN, NDIM)	Dynamic storage management routine: links and unlinks beads on threads of threaded main storage array; maintains linkages in TSM Management Control Tables.

Table 3-6. Simulator Internal Support Routines (Class 4 Subroutines) in the SU File (Sheet 2 of 2)

Subroutine Name	Subroutine Function
LOCATE (TBL,R, INITL,TID,IT, ISTAT,ND)	Chases linkage chains in TSM Control Tables.
MAKE (TBL,R,INDX, INITL,TID,NEW,IT, IER,ND)	Inserts new thread bead on thread of threaded array. Inserts new links in TSM Management Control Tables.
BREAK (TBL,INDX, INITL,IT,ND)	Deletes bead from thread of threaded main storage array. Deletes links in TSM Management Control Tables.
ENSRT (TIME,EVNT, RNODE,ISIG,IREC, JREC,KREC)	Inserts new properly positioned events in TSM Event Queue Table.
FERR1 (N)	Outputs Control Table or Event Queue Table error condition message to EMM file and terminates TSM execution.

3.3.2 Description of ICSSM Applications Library Maintenance Program

The ICSSM system Applications Library Maintenance/Update (MU) Subsystem comprises a single program - the LMU program. The structure of the MU Subsystem is shown in figure 3-25.

The MU subsystem provides facilities that allow the ICSSM Librarian/Custodian to modify the contents of the files comprising the LDAG.

The MU subsystem employs a Librarian/Custodian-prepared update (LMU) file. Execution of the Library Maintenance/Update Utility (LMU) program results in appropriate modifications of the contents of the files of the LDAG. Actual insertion of the Application Module in the LM file is done via the host computer public-file management system.

The LMU program operation:

(a) changes the "Number of Modules" field in the appropriate Chapter record of the LC file.

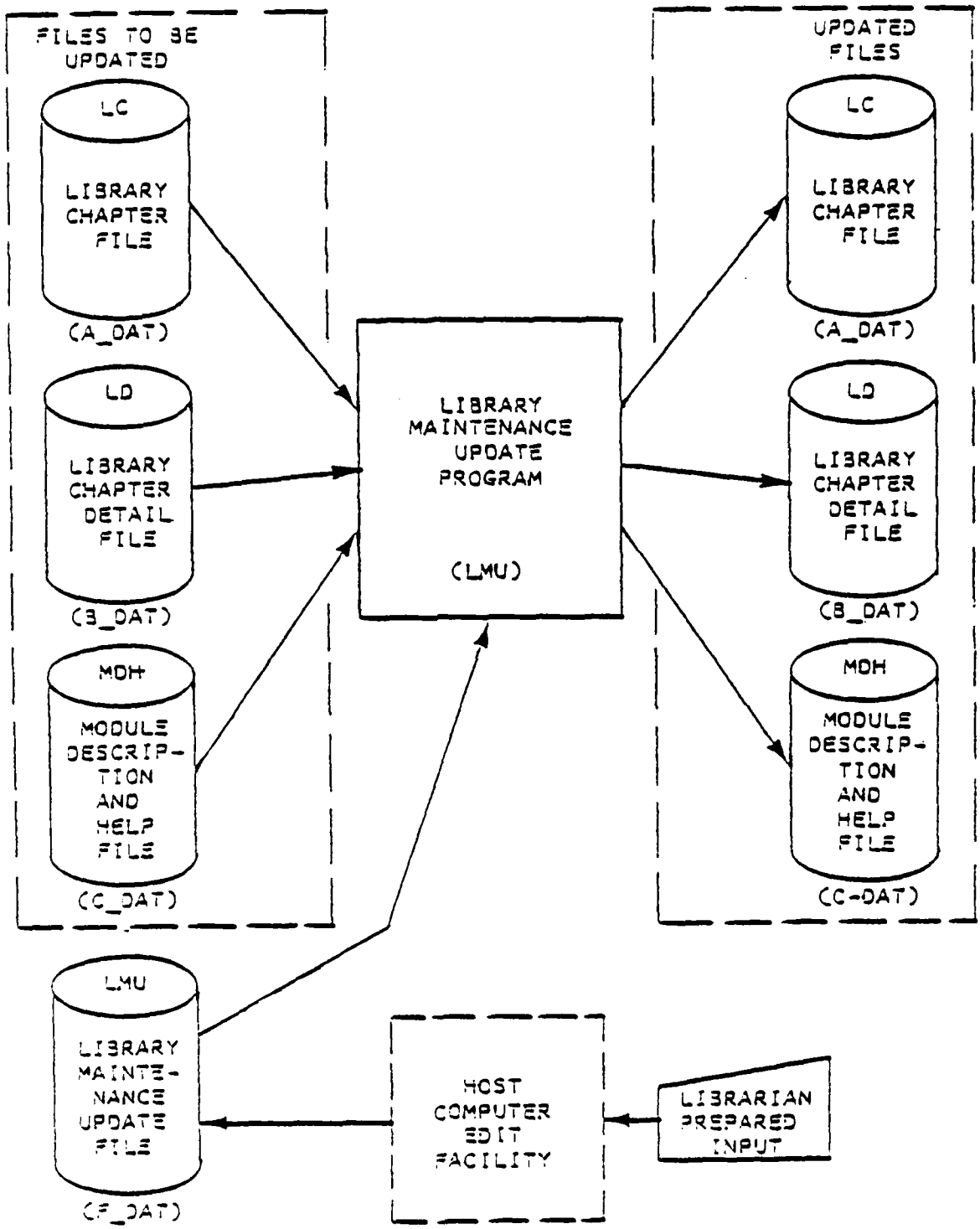


Figure 3-35. Structure of Maintenance/Update (MU) Subsystem

(b) adds Chapter detail records in the LD file to reflect Applications Library (ie, LC file and MDH file) additions

(c) adds the 21 records needed to register an applications module in the MDH file

The inputs to the LMU program are the LC file, the LD file, the MDH file, and the LMU file. The LMU file is a sequential ASCII file. It is prepared by ICSSM host computer edit and file-build utilities. Descriptions of Data Items for the LMU file are provided in table 3-7.

Table 3-7. Formats for LMU File Record Data Items

Item Name	Item Description	Item Format
N	Number of Modules	I3
MODN	Module Name	3A2
CHAPR	Chapter Reference Number	I3
NPAR	Number of Parameters	I2
NHIDP	Number of Hidden Parameters	I4
MTEXT	Module Description Text (3 Records)	3(80A1)
NUMSBR	Number of Subroutines used by Module	I2
PIODES	Parameter, Input Port, Output Port Descriptions	3A6
NUMIP	Number of Input Ports	I1
NUMOS	Number of Output Ports	I1

The output of the LMU program are updated versions of the LC, LD, and MDH files.

SECTION 4

PRODUCT AND RESULTS OBTAINED

This section summarizes the general capabilities of the ICSSM system and discusses the system performance disclosed by the test results.

4.1 ICSSM SYSTEM AND SUB-SYSTEM DESCRIPTIONS

ICSSM is comprised of two components when viewed from a user or application point of view: (1) a Simulation Component (SC), and (2) an Applications Library Component (ALC).

Briefly, the SC provides: (1) input and model formulation capability to a prospective User/Analyst (U/A) via an interactive interface; (2) control and housekeeping facilities needed to carry out simulations; and (3) output data reduction/display facilities for recording or examining simulation results.

Briefly, the ALC contains algorithmic implementations of: (1) communication system functional elements; (2) communications system test equipment or test/measurement methods. These are arranged so that they may (through agencies of the SC) be incorporated into communications systems models as formulated by a U/A, be exercisted together, and the results recorded.

The ICSSM system is employed to configure and exercise a simulation model representing a communications system under study. The ICSSM system oversees the configuration and exercise of the simulation model and the data reduction and display of results produced by the simulation. The ICSSM operates on the RADC Honeywell H-6180/MULTICS computer system for use by RADC U/A.

4.2 TEST RESULTS

The System Function tests demonstrated the applicability and flexibility of ICSSM as a simulation tool, and the ease with which ICSSM may be used by a communications analyst.

The results of the validation tests for Synchronized Spread-spectrum Data Mode (SSSDM) and DCS Quadrative Modem (DCSQM) Models are summarized in tables 4-1 and 4-2 along with theoretical performance.

4.2.1 Analysis of Validation Results - SSSDM Model

The justification for using the theoretical performance for DPSK demodulation in Gaussian noise as a standard with which to compare the performance of the SSSDM model is based upon the central limit theorem [60]. The output of the module SYNCMF is the sum of a number NTSBTP, of independent random variables, each having a near-Gaussian distribution. Thus, the random process at the SYNCMF output converges to a Gaussian process in proportion to the number of independent processes added. The value of NTSBTP = 36 chosen in SSSDM validation is a compromise between a number "sufficient" to justify invoking the central limit theorem, but not so large that it prolongs each validation run beyond a time needed to demonstrate performance. The point of the validation tests is to demonstrate accuracy and effectiveness of ICSSM Applications Library modules, and to verify the speed of simulation by benchmarking a theoretically tractable communication system design. Using larger values of NTSBTP does not enhance the model's ability to verify these properties.

The central limit theorem predicts that the mean of the SYNCMF output will be the signal peak amplitude, SIGVP, multiplied by NTSBTP, when measured at the moment of synchronization. It also predicts that the variance of the SYNCMF output under these conditions is the sum of the individual variances of the Gaussian variates describing the individual chips integrated in the SYNMCF module.

Table 4-1. Summary of Results of SSSDM Model Validation

S/N γ, (dB)	# Data Bits	# Errors	PE Theoretical	PE Measured
-4	311	112	0.3357	0.36
	259	92		0.3552
-2	233	62	0.2660	0.2660
	259	68		0.2625
0	311	65	0.1839	0.2090
	311	52		0.1672
	1039	202		0.1944
2	311	26	0.1024	0.0836
	259	18		0.0694
	1039	115		0.1106
4	259	5	0.0405	0.0193
	5409	193		0.0356
6	779	8	0.9×10^{-2}	1.0269×10^{-2}
	5409	40		0.7395×10^{-2}
	10919	37		0.3388×10^{-2}
8	10919	5	0.909×10^{-3}	0.4579×10^{-3}
	10919	3		0.2747×10^{-3}

Table 4-2. Summary of DCSQM Validation Tests

SNR (dB)	SNR	No. of Data Bits	No. of Errant Bits	Theoretical P_E	Measured P_E
1	1.2589	101×10^3	39772	0.365	0.393
4	2.5118	101×10^3	29431	0.266	0.291
6	3.981	1001×10^3	198478	0.184	0.198
8	6.309	1001×10^3	104224	0.103	0.104
10	10.0	1001×10^3	37305	0.041	0.037
13	20.0	1001×10^3	2308	3.36×10^{-3}	2.30×10^{-3}
13.95	24.858	4×10^6	2458	10^{-3}	6.145×10^{-4}

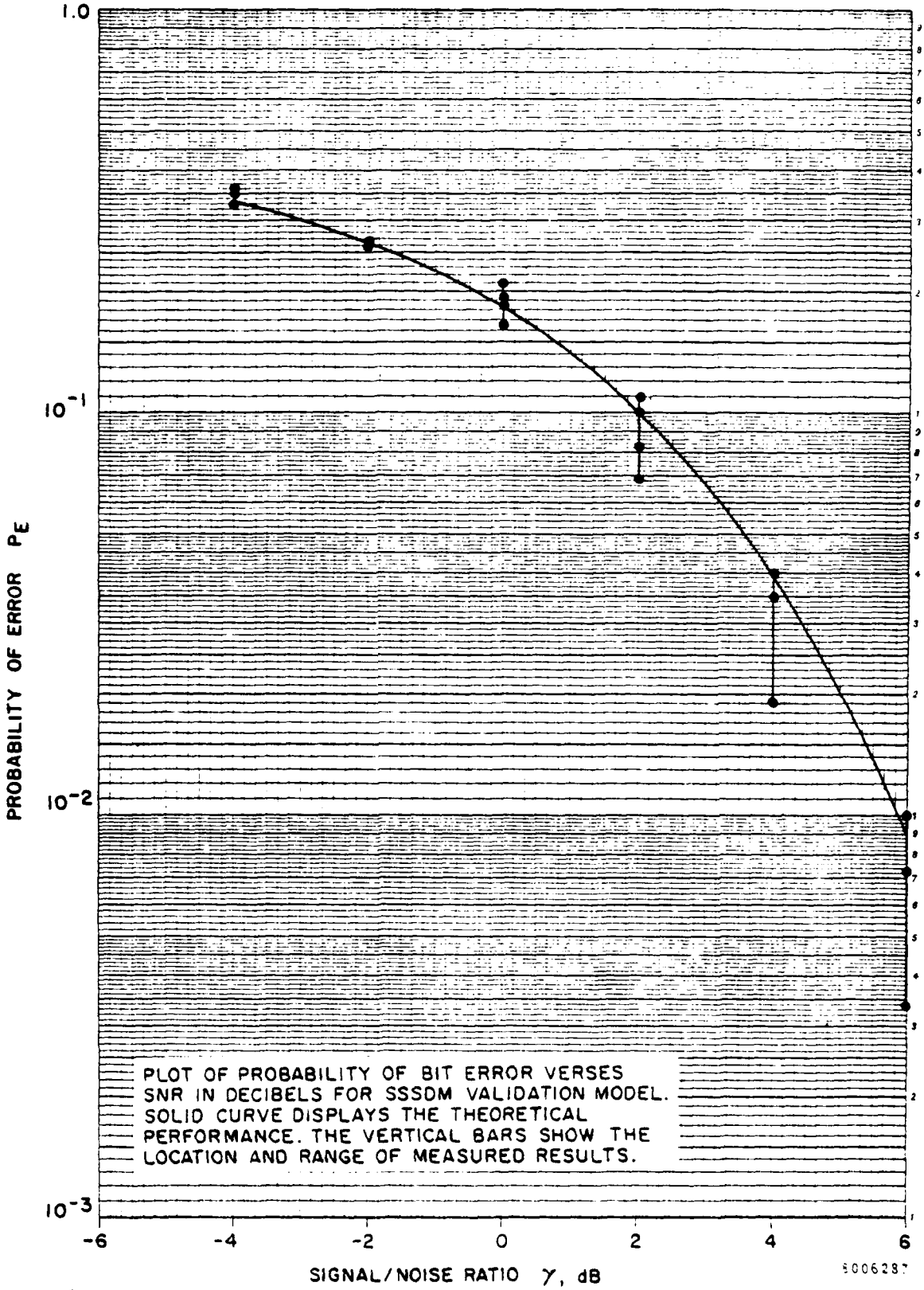


Figure 4-1. SSSDM Probability of Bit Error Vs SNR

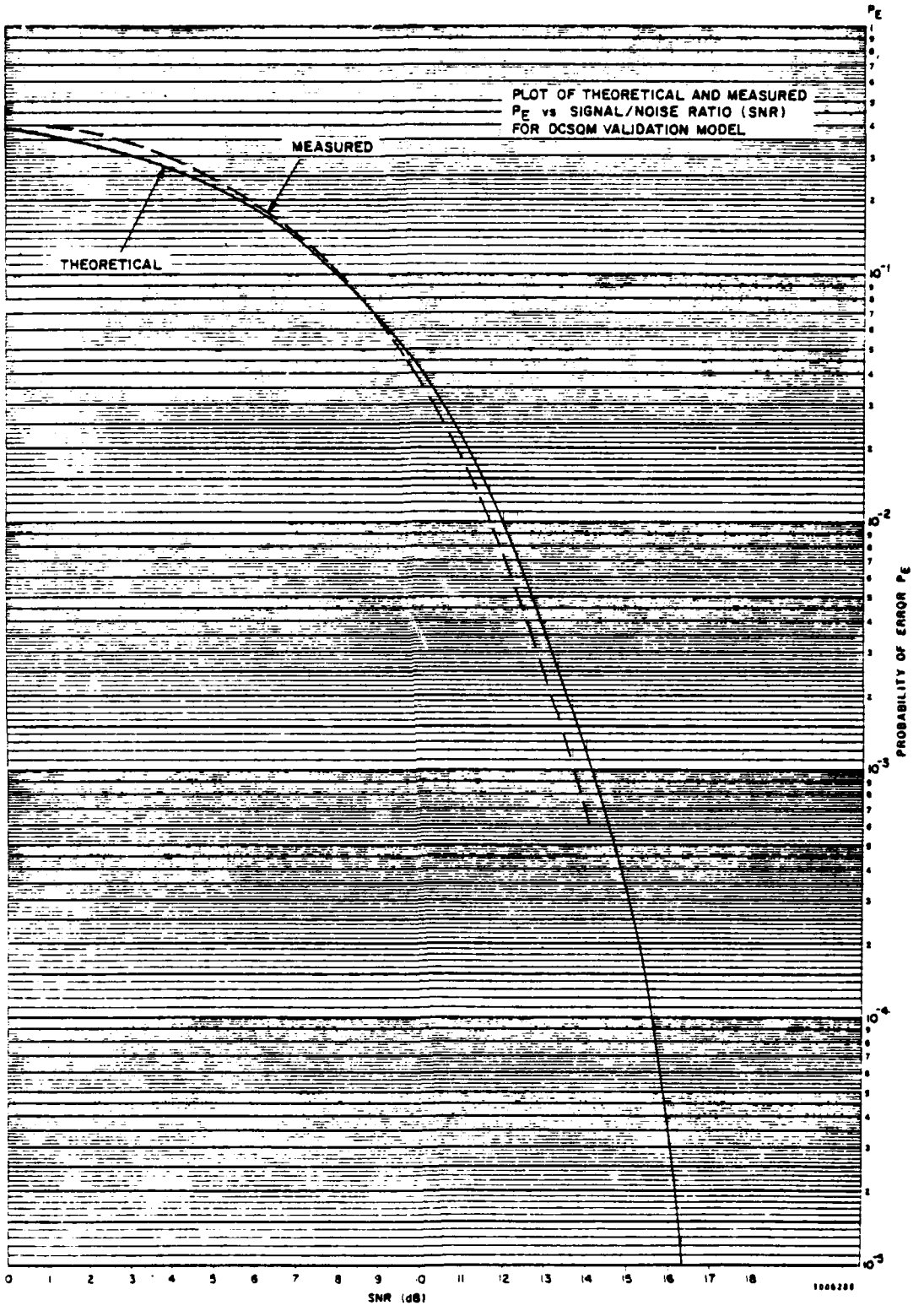


Figure 4-2. DCSQM Probability of Bit Error Vs SNR

The comparison of measured and theoretical P_E values (see figure 4-1 and table 4-1) shows a performance close to theoretical for SNR up to about +4 dB, where a difference of 12% between measured and theoretical results was observed. Results for SNR of +6 dB were in error by 45%. The experiments of the +6 dB and +8 dB had insufficient sample sizes (85 and 8 respectively) to have a high confidence of being close to the actual model mean error. Further, the error rate in these regions of operation depend upon the exact shape of the noise distribution (see figure 4-3) in the tails beyond 3.50 units from the mean value. Two effects distort the distribution in this region: (1) the truncation in simulation noise voltage values due to the finite size of the computer representation for real numbers and (2) the "swelling up" of the distribution near the mean value compared with the "thinning out" of the distribution in the tails, in order that the total area under the computer-generated distribution curve be equal to unity. Both these effects act to cause an improvement in performance compared to theoretical. The measured results fall on the "improved" side of the theoretical performance curve. Only about 0.04% of the area of the normal curve lies beyond about 3.50 units from the mean, and it is in this region that the performance at SNR of +6 dB and greater lies.

4.2.2 Analysis of Validation Results - DCSQM Model

The comparison of measured and theoretical P_E values (figure 4-2 and table 4-2) shows measured performance close to theoretical for QPSK modulation. For SNR values from +1 to +10 dB a difference of less than 10% between measured and theoretical results was observed, with a measured value at SNR of +8 dB within 1% of theory. Results for SNR values of 13 dB and 13.95 dB were in error by more than 30%. The measured error rate in these regions of operation depends on the exact shape fo the noise distribution in the tails beyond 3.5σ from the mean value. Test analysis for the

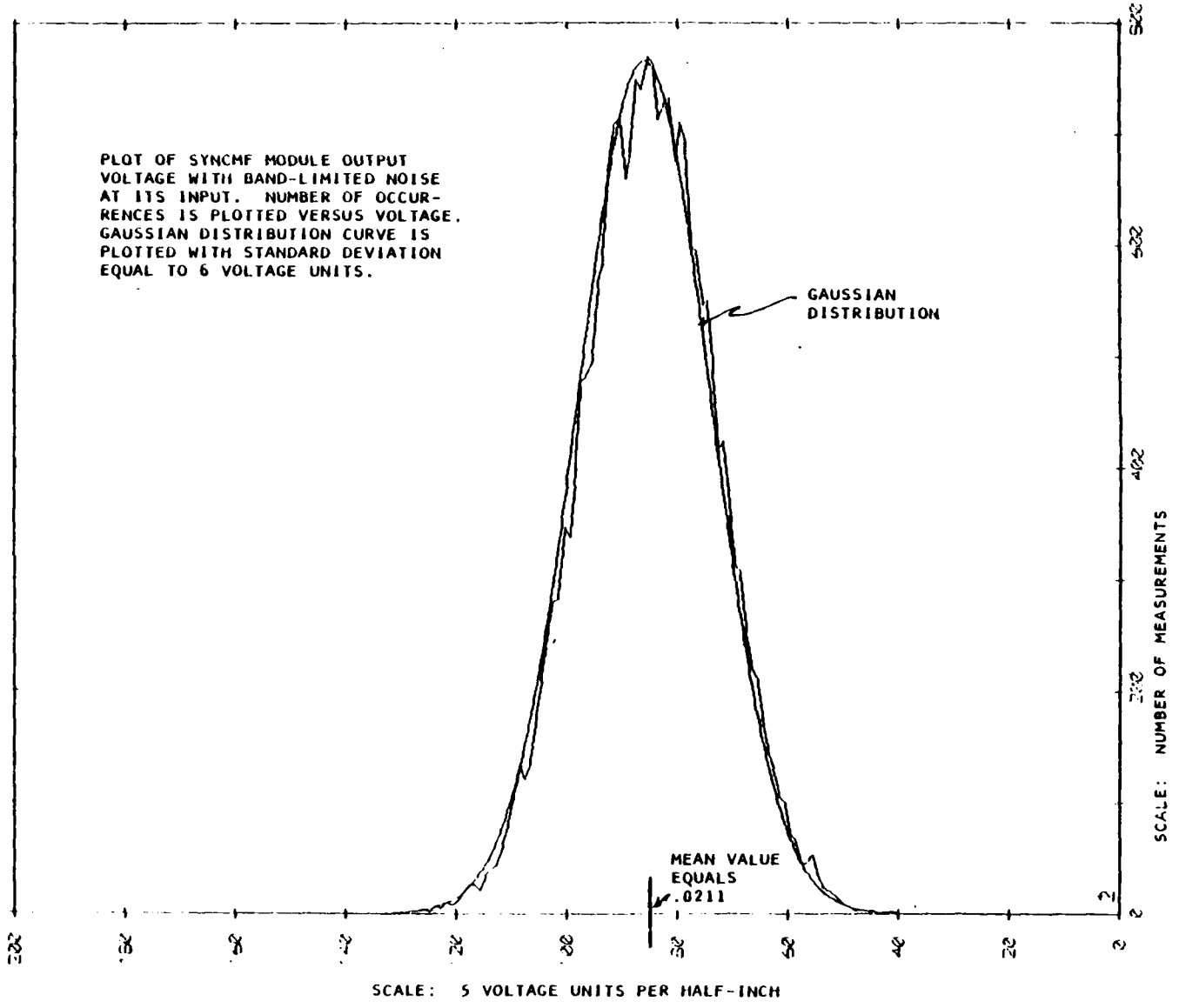


Figure 4-3. Matched Filter Output Voltage

SSSDM model has already shown that the noise distribution is distorted in this region, the effect being an improvement in measured performance compared to theoretical.

4.3 DEMONSTRATED CAPABILITIES

The Function Tests and Validation Tests have demonstrated that the ICSSM system is applicable to the non-real-time modeling and simulation of point-to-point line-of-sight communications systems.

The Tests have demonstrated that the ICSSM system is expandable in that Applications Library elements (ie, modules and DMS) may be added to address new areas of simulation.

The Tests have demonstrated that modeling and simulation results derived from ICSSM are accurate within the limitations imposed by specific computer host hardware and software, and posed by utility programs available or provided (eg, DMS RPIGRG adopted from a random generator designed by Rennselaer Polytechnic Institute and providing the best of four Gaussian random generators tested) for simulation.

The Tests demonstrate the speed with which simulations may be executed and the speed with which substantial amounts of difficult-to-calculate data may be obtained through services provided by the ICSSM system.

4.4 VALIDATION MODEL TIMING

The simulation timing data for the validation tests for SSSDM and DCSQM models run on the RADC Honeywell/MULTICS computer system are summarized in tables 4-3 and 4-4. The SSSDM model timing data is typical of a model which performs numerous and complex processing, whereas the DCSQM model timing data is typical of a model which performs simpler processing. The data show typical CPU resources needed for ICSSM models. The data show that the CPU time per data bit decreases as the total number of data bits increases. This is due to the fixed overhead, for initialization, file I/O, etc, for ICSSM system. The total CPU time versus the number of data bits is plotted for the SSSDM model in figure 4-4, and for the DCSQM model in figure 4-6. The CPU-seconds-per-bit versus the number of data bits is also plotted for the SSSDM model in figure 4-5, and for the DCSQM model in figure 4-7.

Table 4-3. Summary of SSSDM Model Timing

# Data Bits	Total CPU Seconds	CPU Seconds Per Data Bit
207	675	3.26
259	831	3.208
311	937	3.01
1039	2972	2.86
5409	15030	2.778
10919	30240	2.767

Table 4-4. Summary of DCSQM Model Timing

# Data Bits	Total CPU Seconds	CPU Seconds Per Data Bit
30,000	267	8.9×10^{-3}
101,000	890	8.8×10^{-3}
1,001,000	8760	8.75×10^{-3}
4,000,000	34810	8.7×10^{-3}

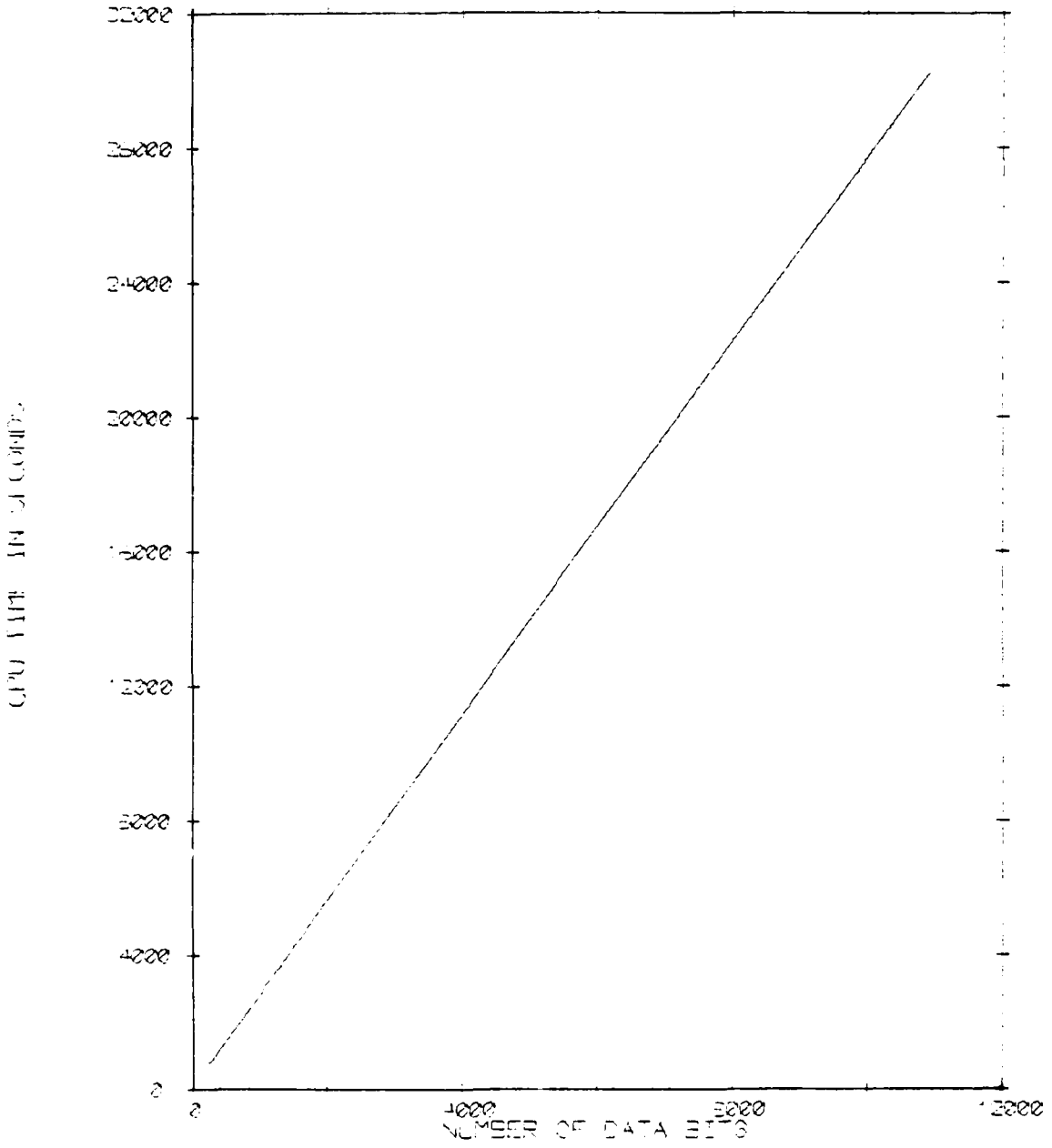


Figure 4-4. SSSDM CPU Timing Vs Number of Data Bits Simulated

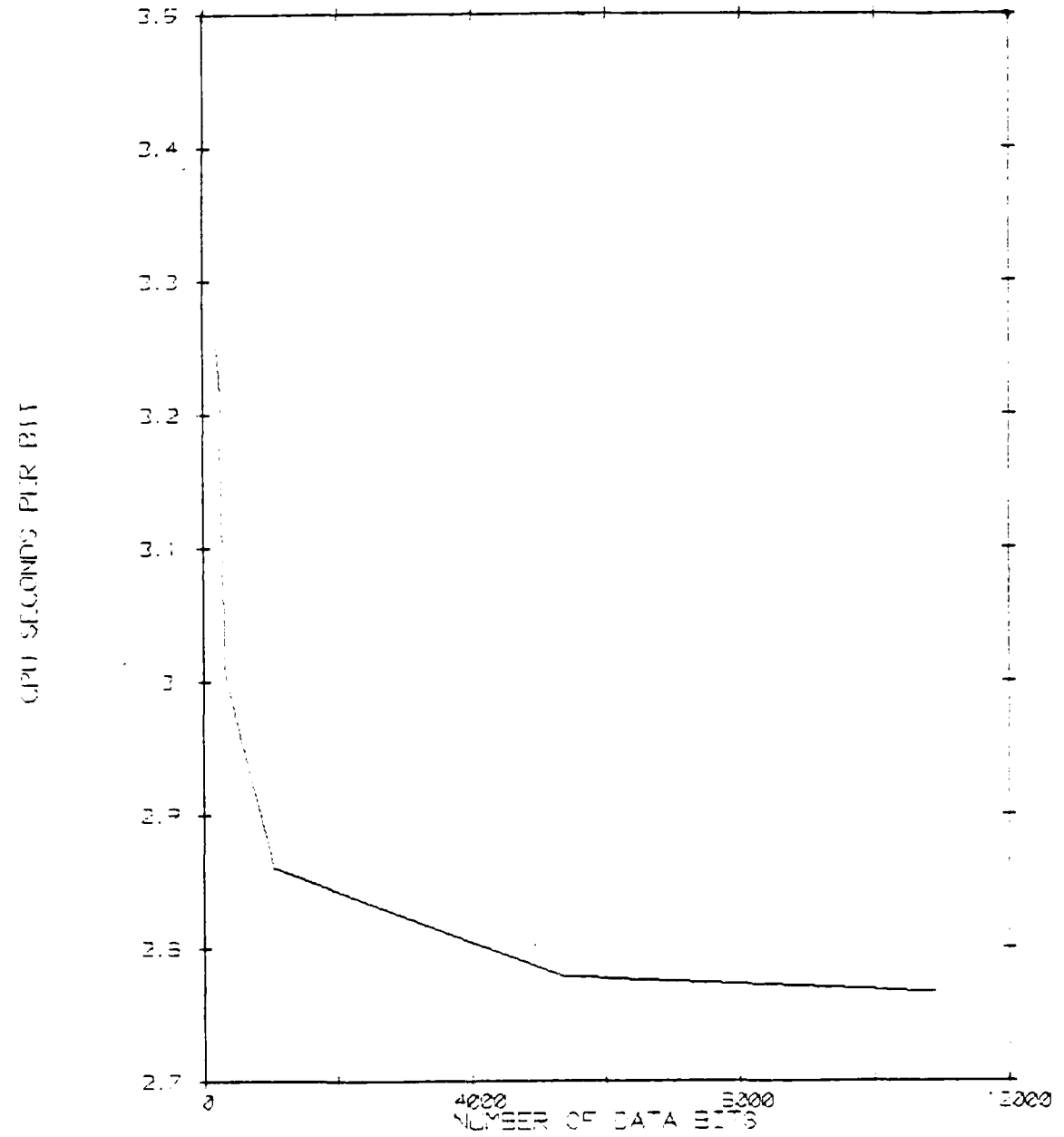


Figure 4-5. SSSDM CPU Time Per Bit Vs Number of Data Bits Simulated

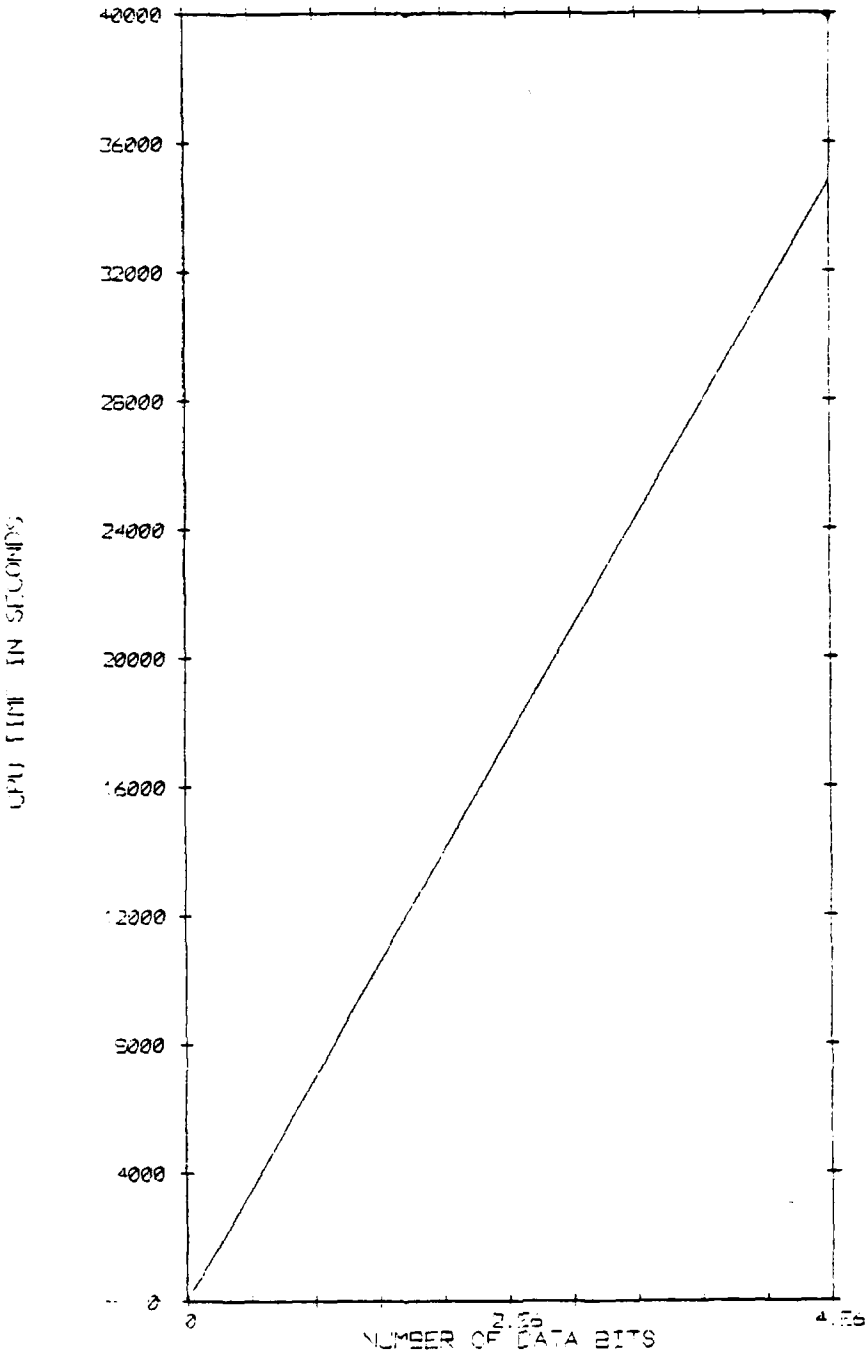


Figure 4-6. DCSQM CPU Timing Vs Number of Data Bits Simulated

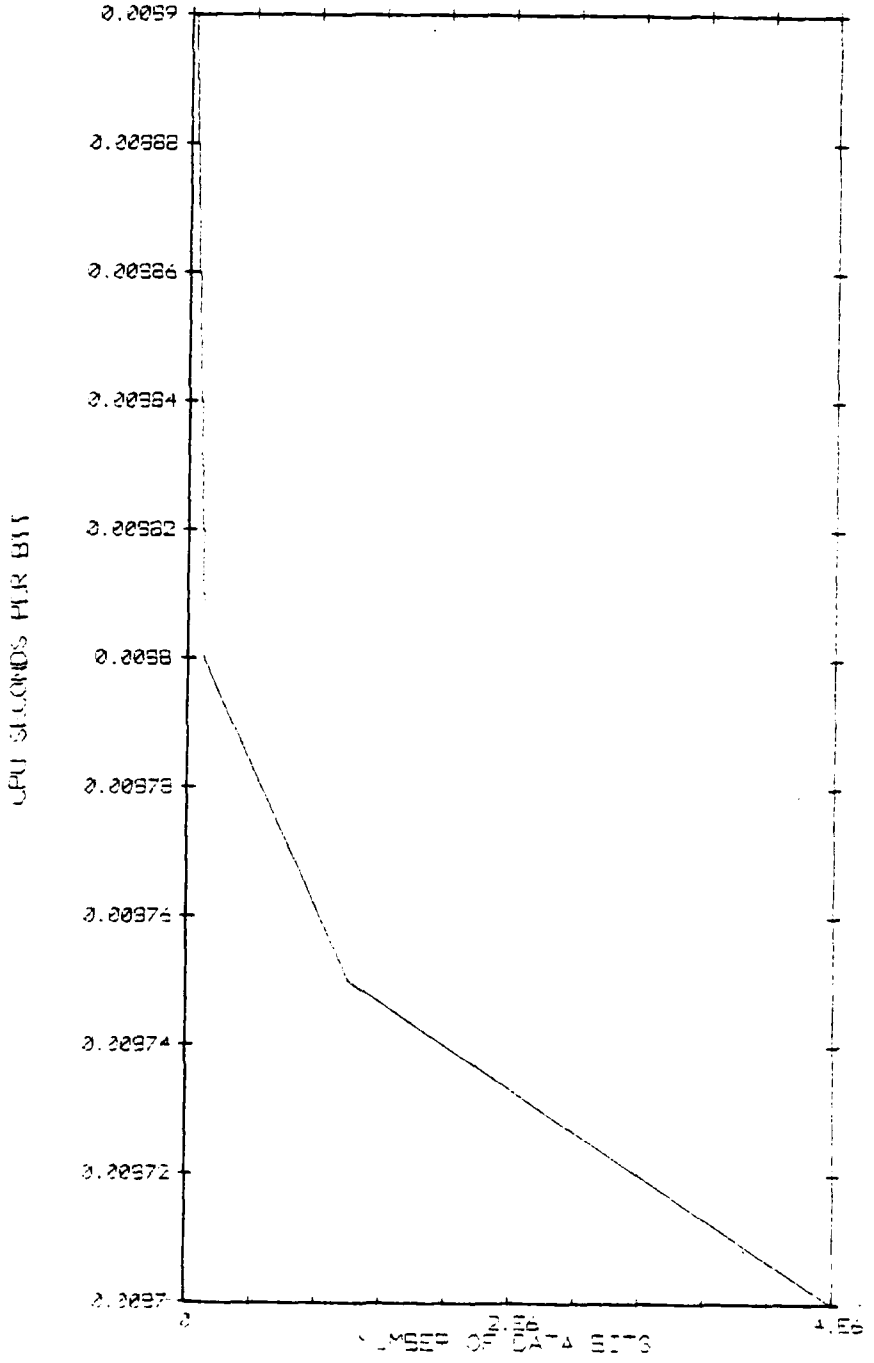


Figure 4-7. DCSQM CPU Time Per Bit Vs Number of Data Bits Simulated

SECTION 5

RECOMMENDATIONS AND CONCLUSIONS

The results reported on and discussed in Section 4, and general experience with the operation of the Initial ICSSM System, prompt the recommendations outlined below. These recommendations fall into two categories: those pertaining to operations of the ICSSM facility, and those pertaining to enhancements and improvement to the ICSSM structure. A summary conclusion is also included.

5.1 RECOMMENDATION PERTAINING TO ICSSM OPERATION

Recommendations which address the efficiency of the ICSSM system in actual use are discussed in this paragraph.

5.1.1 Timings and Simulation Speed

The complex interrelationships between the need for flexibility and generality in ICSSM on the one hand, and the need for speed and ease-of-use on the other hand suggest that ICSSM simulation executions be studied using an in-core trace/timing software utility. This will provide a profile of where (what portions of computer code) the typical simulation spends most of its time. This would be useful to determine where improvements and "tightening" of code would have the greatest impact on speed and simulation efficiency, while ensuring that more arbitrarily selected sites for coding improvement would not exact a penalty in reduced generality and flexibility without a justification in terms of trade-off with speed and efficiency.

5.1.2 Accuracy and Random Processes

The accuracy of the simulations under ICSSM is almost totally dependent on the design of algorithms for use in Applications Library modules. Yet, fundamentally any process requiring the modeling of "noise" or random effects should rely on adequate random process generations. Without these, accuracy

is impaired. This suggests that study and design of better random generators for ICSSM use be pursued. Further, on-line generation of random processes is time consuming. This could be avoided if random data were prepared off-line for use during simulations. This suggests that a study and design of efficient and accurate general methods for preparing and using pre-simulation-generated random data of requisite fidelity be pursued.

5.1.3 Overhead Reduction in Simulation

The Applications Library module designer has a strong interest in design for general use. The re-entrant philosophy for ICSSM Applications Library module design helps in this direction. However, the desires to generalize and to isolate communications - theoretic "unit" processes in individual modules results in the tendency to design and employ many separate modules in configuring a TSM. Each module employed in a TSM represents an individual passage of control and data to and from the simulator executive (ie, the Exercisor Kernal) each time that that module is invoked during a simulation. This passage of control and data represents an "overhead" from the user's point of view. This suggests that a study and analysis be made to determine the minimum overhead possible, and to devise rules of trade-off for designing modules and TSM which approach this minimum and devise means to determine the "optimum size" for Applications Library module design.

One possibility for good trade-off is to design a means whereby individual modules could be conglomerated into a "unit" which could then be employed as a simulation element in its own right. This would allow individual modules to be designed which manifest generic or canonical modeling functions, and yet be combined into a "super-module" or "unit" which interfaces with the simulation executive. Thus, pre-configuration of such "units" prior to set-up and simulation of a "large" model could be a feasible means of finding and implementing the correct trade-off point.

5.1.4 Speed Improvement by Use of Array Processor Adjuncts
The ICSSM system design in part assumes the presence of an array processor peripheral to the host computer. Emulator routines in the ICSSM Applications Libraty provide means to design modules using Array Processor "functions" or "calls" even in the absence of an array processor. Efficient use of the array processor requires the transfer of relatively large data vectors and arrays of data to and from that processor. Requirements on module design and the algorithmic details of particular models constructed for ICSSM simulations, on the other hand, may dictate processing of relatively small vectors and arrays. For this reason an array processor would not significantly increase the speed of some modules or processors. The overall benefit of an array processor to an ICSSM simulation depends on the effectiveness of applying array mainpulations to modeling algorithms. This suggests that a study and analysis to find optimum rules for employing array processor "calls" be pursued. These rules, if formulated, would determine the type and location within modules where array processor manipulation would be optimally effective.

The SSSDM and DCSQM model timings are shown in Section 4.4. These timings are based on validation runs on the RADC Honeywell/MULTICS computer system which does not include an array processor. The models, however, were implemented using array processor emulation routines in the ICSSM Applications Library. It is estimated that an array processor peripheral would improve the run time by approximately 10% for the SSSDM model and 15% for the DCSQM model. The primary time savings would be attributed to the array processor's speed in vector multiplication, addition, and division. This improvement implies that for 10919 Data bits the SSSDM model would require an average of 2.49 CPU seconds per bit, and for 4 million data bits the DCSQM model would require an average of 7.35×10^{-3} CPU seconds per bit. Greater speed enhancements could be achieved by an ICSSM model that made more extensive use of the array processor and used it in an optimum fashion.

5.2 RECOMMENDATIONS PERTAINING TO IMPROVEMENTS IN ICSSM STRUCTURE

Recommendations which address ease-of-use, adaptability, expandability and applicability of the ICSSM system are discussed in this paragraph.

5.2.1 User/System Interface and User Guidance

The translations of communications-analytic formulations into tangible simulation models requires either the design of or the existence of suitable modeling elements and requires the translation of the communications concept to block-diagram form. The requisite amount of skill and experience needed with this process in the ICSSM setting can be reduced by carefully arranged interactive scripts employed in model configuration, and by adequate tutorial treatment of module and model design as required in an ICSSM setting. Experience so far suggests that both these aspects of ICSSM structure be enhanced:

(1) improved interactive dialogue during model configuration should be designed, with particular emphasis on user access to appropriate tutorial/help data during these sessions;

(2) independent tutorial material, both computer-aided and documentary, should be created to introduce, guide and regiment the ICSSM user in all aspects of module design, model configuration and ICSSM simulation control/data management.

5.2.2 Recursive Modeling

When a complete simulation model (TSM) has been configured and completed, the resultant software assemblage is treated as an application program by the host computer operating system, indistinguishable from any other application program. From the the ICSSM point of view, if a TSM could be treated as a subroutine structured according to ICSSM requirements, it <u>itself</u> could be installed in the Applications Library and employed in a more complete ICSSM TSM as if <u>it</u> were a module. By means of this artifice, the ICSSM system could "bootstrap" itself into more and more complicated modeling structures, thus providing a kind of recursive modeling capability. While there

are some practical problems involved (eg, ICSSM TSM read from and write-to particular files, and this I/O programming is embedded in the structure of the Exercisor Kernal, so that difficult file management problems would arise if steps were not taken to mitigate them), this possibility suggests that a study and design to produce a boot-strapping facility be pursued.

5.2.3 Checkpoint/Restart and Simulation-In-Process Monitoring

ICSSM simulations can execute for a long time. During these executions, accidents or computer management decisions could bring about execution abortion, causing a loss of all the computer resources invested in the simulation up to that time. A checkpoint/restart facility should be added to ICSSM to neutralize these losses.

Further, ICSSM should be modified to provide means to monitor the progress of the simulation, so that cogent decisions as to whether to continue execution or to abort can be made. Coupled with a checkpoint/restart capability, this would vastly improve resource management of the ICSSM host computer.

5.2.4 Quick-Look Capabilities

Simulations can take a long time to run to completion. Simulation models are relatively difficult to design and configure even with the aid of a system like ICSSM. Uncertainty always exists at the outset of a simulator execution as to whether the model is valid. Means should be provided within ICSSM to make a quick appraisal of the simulator output data (and all intermediate data as well) to determine if the model is executing correctly before large amounts of computer resources are invested. Such a "quick-look" facility would be a worthwhile enhancement to the ICSSM system.

5.2.5 Output Data Suppression

Currently, ICSSM generates and stores output data from every Library module incorporated into a TSM. This has been observed to require large amounts of computer storage. Some of the output data might never be studied by the user once he has determined that the simulator is executing correctly, say by means of a short test or observation run (refer also to paragraph 5.2.4). This suggests that ICSSM be modified to provide means to suppress all but the significant output data from a simulation, the suppression control being provided to the user during model configuration/definition.

5.3 CONCLUSION

Experience with the initial ICSSM shows that it is a viable, worthwhile, powerful means of providing simulation-analytic insights and solutions/results for difficult communication system design and analyses problems. Its further development and refinement are strongly suggested.

APPENDIX A

REFERENCES

[1] R. W. Moss, J. L. Hammond, E. E. Donaldson, "Interactive Communication Systems Modeling Study," Interim Technical Report, RADC TR-76-155, May 1976, A026409.

[2] R. W. Moss, R. W. Rice, D. R. Sentz, "Interactive Communication Systems Modeling Study," Final Technical Report RADC TR-77-42, February 1977, ADA037833

[3] Losson, Thomas R., Danial D. McRae, "Broadband Digital Modem," Rome Air Development Center, Final Technical Report, RADC TR-76-117, May 1977, ADA025399

[4] F. M. Torre, "Adaptive Interference Suppression Correlation Processor," Draft of Final Report submitted to RADC September 29, 1977 (Contract F30602-76-C-0219). A073357.

[5] Gilbert, E. N., "Capacity of a Burst-Noise Channel," BSTJ, vol. 39, p. 1253, 1960.

[6] Elliott, E. O., "A Model of the Switched Telephone Network for Data Communications," BSTJ, vol. 44, pp. 89-110, 1965.

[7] Berger, J. M., and B. Mandelbrot, "A New Model for Error Clustering in Telephone Circuits," IBM J. Res. & Dev., vol. 7, p. 224, 1963.

[8] Sussman, S. M., "Analysis of the Pareto Model for Error Statistics on Telephone Circuits," IEEE Trans. on Communication Systems, col. CS-11, p. 213, 1963.

[9] V. K. Prabhu, "Error Rate Considerations for Coherent Phase-Shift Keyed Systems With Co-channel Interference," Bell Syst. Tech. J., vol. 48, no. 3, pp. 743-767, March 1969.

[10] A. S. Rosenbaum, "Binary PSK Error Probabilities With Multiple Co-channel Interferences," IEEE Trans. Commun. Tech., vol. COM-18, pp. 241-253, June 1970.

[11] J. Goldman, "Statistical Properties of a Sum of Sinusoids and Gaussian Noise and its Generalization to Higher Dimensions," Bell Syst. Tech. J., vol. 53, no. 4, pp. 557-580, April 1974.

[12] R. Fang and O. Shimbo, "Unified Analysis of a Class of Digital Systems in Additive Noise and Interference," IEEE Trans. Commun., vol. COM-21, pp. 1075-1091, October 1973.

[13] S. Benedetto, E. Giglieri, and V. Castellani, "Combined Effects of Intersymbol, Interchannel, and Co-channel Interference in M-ary CPSK Systems," IEEE Trans. Commun., vol. COM-21, pp. 997-1008, September 1973.

[14] M. C. Jeruchim and F. E. Lilley, "Spacing Limitations of Geostationary Satellites Using Multilevel PSK Signals," IEEE Trans. Commun., vol. COM-20, October 1972.

[15] A. S. Rosenbaum, "PSK Error Performance Gaussian Noise and Interference," Bell Syst. Tech. J., vol. 48, no. 2, pp. 413-442, February 1969.

[16] V. K. Prabhu, "Error Probability Upper Bound for Coherently Detected PSK Signals with Co-channel Interference," Electron., Lett., vol. 5, no. 16, pp. 383-386, August 1969.

[17] J. M. Aein, "On the Effects of Undesired Signal Interference to a Coherent Digital Carrier," Inst. for Defense Analyses, Paper P-812, February 1972.

[18] J. M. Aein and R. D. Turner, "Effect of Co-channel Interference on CPSK Carriers," IEEE Trans. Commun., vol. COM-21, pp. 783-790, July 1973.

[19] A. S. Rosenbaum and F. E. Glave, "An Error Probability Upper Bound for Coherent Phase-Shift Keying with Peak Limited Interference," IEEE Trans. Commun., vol. COM-22, pp. 6-16, January 1974.

[20] F. E. Glave and A. S. Rosenbaum, "An Upper Bound Analysis, for Coherent Phase-Shift Keying with Co-channel, Adjacent-Channel, and Intersymbol Interference," IEEE Trans. Commun., vol. COM-23, pp. 586-597, June 1975.

[21] J. Krishnamurthy, "Bounds on Probability of Error Due to Co-channel Interference," IEEE Trans. Aerosp. Electron. Syst., vol. AES-11, pp. 1373, November 1975.

[22] M. J. Wilmut and L. L. Campbell, "Signal Detection in the Presence of Co-channel Interference and Noise," IEEE Trans. Commun., vol. COM-20, pp. 1153-1158, December 1972.

[23] J. Goldman, "Detection in the Presence of Spherically Symmetric Random Vectors," IEEE Trans. Inform. Theory, vol. IT-22, pp. 52-59, January 1976.

[24] G. A. Miller and J. C. R. Lickleder, "The Intelligibility of Interrupted Speech," p. 167, JASA, vol. 22, 1950.

[25] K. D. Kryter, "Methods for the Calculation and Use of the Articulation Index," p. 1695, JASA, vol. 34, November 1962.

[26] E. J. Gumbel, "Statistics of Extremes," New York Columbia Univ., Press 1958.

[27] S. B. Weinstein, "Theory and Application of Some Classical and Generalized Asymptotic Distributions of Extreme Values," IEEE Trans. Inform. Theory, vol. IT-19, pp. 148-154, March 1973.

[28] Moss, R. W., R. W. Rice, D. R. Sentz, "Interactive Communication Systems Modeling Study," Rome Air Development Center, RADC-TR-42, Feb 1977. ADA 037833.

[29] Shannon, Robert E., "Simulation Modeling and Methodology," Bicentennial Winter Simulation Conference Proceedings, Vol 1, pp 9-15, December 1976.

[30] RADC Computer Software Development Specification, Specification No. CP 0787796100B. Code Ident 07877, Rome Air Development Center, N.Y., 28 February 1977.

[31] Maynard, Denis R., "Introduction to the RADC R&D Computer Facility," Rome Air Development Center, RADC-TR-77-61, March 1977. ADA038657.

[32] McEvoy, J. B., N. J. Sturdevant, "RADC's Digital Communications Experimental Facility," AFCEA, Signal Magazine, Vol XXVIII, No. 10, July 1974, pp. 4-12.

[33] Moss, R. W., R. W. Rice, P. K. Leong, "Model for Interactive Design and Analysis of Communication Systems," Bicentennial Winter Simulation Conference Proceedings, Vol 1, pp. 185-189, December 1976.

[34] Clema, J., F. Scarpino, "A General Purpose Tool for Interactive Simulations" Bicentennial Winter Simulation Conference Proceedings, Vol 2, 475-484, December 1976.

[35] "Digital Transmission System Design," DCEC TR-3-74.

[36] "DCS Digital Transmission System Performance," DCEC TR-12-76.

[37] DELETED

[38] DELETED

[39] Bello, Philip A., et al, "Line-of-Sight Techniques Investigation," Rome Air Development Center, Final Report, RADC-TR-74-330, January 1975, ADA 006104.

[40] "Adaptive Modulation and Error Control Techniques," IBM Corp, Contract AF30(602)-3603, RADC-TR-66-169, May 1966. AD484188.

[41] Schmandt, Frederick D., "DCA Systems Evaluation - Phase 1," Rome Air Development Center, In-house Report, RADC-TR-76-358, February 1977, ADA038607.

[42] Losson, Thomas R., Daniel D. McRae, "Broadband Digital Modem," Rome Air Development Center, Final Technical Report, RADC TR-76-117, May 1977, ADA025399.

[43] Jones, Norman G., et al, "Microwave Data Transmission Test Program - QPSK Modems," Rome Air Development Center, Final Report, RADC TR-74-274, November 1974, ADA002840.

[44] DELETED

[45] DELETED

[46] DELETED

[47] DELETED

[48] DELETED

[49] DELETED

[50] "Functional Description for Interactive Communication System Simulation Model (ICSSM)",(U), Hazeltine Corp. Report #6371 Dec. 1979.

[51] "System Specification for Interactive Communications System Simulation Model (ICSSM)",(U), Hazeltine Corp. Report #6382 Dec. 1979.

[52] "Programming Specifications for Interactive Communication System Simulation Model (ICSSM)",(U), Hazeltine Corp. Report #6387 Feb. 28, 1980.

[53] "User/Analyst Manual for Interactive Communication System Simulation Model (ICSSM)",(U) Hazeltine Corp. Report #6390 Jan. 28, 1980.

[54] "Test Analysis Report on Interactive Communication System Simulation Model",(U), Hazeltine Corp. Report #6405 April 1980.

[55] "Program Maintenance Manual for Interactive Communication System Simulation Modeling (ICSSM) System," (U), Hazeltine Corp. Report #6397, June 1980.

APPENDIX B

LIST OF ABBREVIATIONS, ACRONYMS, AND SYMBOLS

The abbreviations and acronyms used in this document are listed below. This list does not include abbreviations and acronyms that are in accordance with MIL-STD-12.

ALC	Application Library Component
CTMP	Control Table Management Processor
CW	Coefficient Work
DCLF	Digital Communications Experimental Facility (DICEF)
DCS	Defense Communication System Section
DMI	Direct Matrix Inversion
DMS	Dependent Modeling Subroutines
EK	Exercisor Kernal
EMM	Error Message and Module
EQP	Event Queue Table Processor
ES	Exercisor/Simulator
EVJ	Event Journal
EXJ	Execution Journal
FCBA	FORTRAN Common-Block Alignment
FMD	FORTRAN Model Description
GDU	Graphics and Display Utilities
ICSSM	Interactive Communications System Simulation Model
IMS	Intermediate Model Specification
LC	Library Chapter
LD	Library Detail
LDAG	Library/Directory Applications Group
LDUG	Library/Directory Utilities Group
LM	Library Module
LMU	Library Maintenance Update
MC	Model Configuration
MCP	Model Configurator Pre-Compiler
MCS	Model Configuration Select
MDH	Module Description and Help
MOP	Module and Output Port
MTE	Model Table Extract
MU	Maintenance/Update
PP	Post-Processor
PM	Process Module
PPE	Post-Processor Exercisor
PPF	Post-Processor Function
PPL	Post-Processor List
PPS	Post-Processor Selector

RADC	Rome Air Development Center
SAE	Systems Analysis and Evaluation
SC	Simulation Component
SFC	Set FORTRAN Common
SIG	Signal Output
SSL	Scientific Subroutine Library
SU	Support Utilities
SW	Signal Work
TSM	Target Simulation Model
U/A	User/Analyst

MISSION
of
Rome Air Development Center

RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control Communications and Intelligence (C^3I) activities. Technical and engineering support within areas of technical competence is provided to ESD Program Offices (POs) and other ESD elements. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.


LME